United States Patent [19]
Bourgeois et al.
[11] Patent Number: 5,060,170
[45] Date of Patent: Oct. 22, 1991

[54] SPACE ALLOCATION AND POSITIONING METHOD FOR SCREEN DISPLAY REGIONS IN A VARIABLE WINDOWING SYSTEM

[75] Inventors: Nancy E. Bourgeois, Cary; Sandra L. Hause, Raleigh; Arwin B. Lindquist, Cary, all of N.C.

[73] Assignee: International Business Machines Corp., Armonk, N.Y.

[21] Appl. No.: 391,290

[22] Filed: Aug. 9, 1989

[51] Int. Cl.[5] G06F 3/14

[52] U.S. Cl. 364/521; 364/518; 340/721

[58] Field of Search 364/518, 521; 340/721, 340/734, 731, 750, 798, 799

[56] References Cited

U.S. PATENT DOCUMENTS

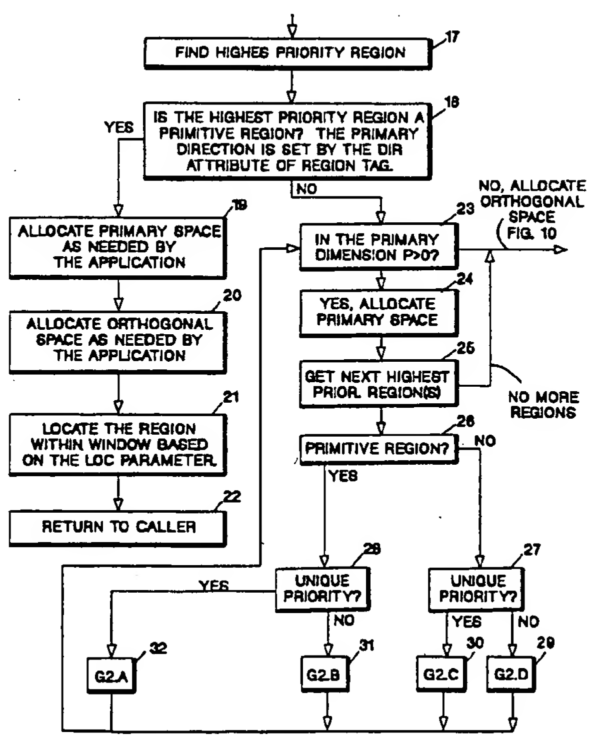
4,598,384	7/1986	Shaw et al.	364/521 X
4,651,146	3/1987	Lucash et al.	340/750 X
4,653,020	3/1987	Cheselka et al.	340/747 X
4,663,617	5/1987	Stockwell	340/726
4,698,779	10/1987	Holden et al.	364/520
4,731,606	3/1988	Bantz et al.	340/721 X
4,783,648	11/1988	Homma et al.	340/721 X
4,789,962	12/1988	Berry et al.	364/521 X
4,794,386	12/1988	Bedrij et al.	364/521 X
4,823,108	4/1989	Pope	364/521 X
4,823,303	4/1989	Terasawa	364/521
4,961,070	10/1990	Maher et al.	364/521 X
5,001,697	3/1991	Torres	340/721 X

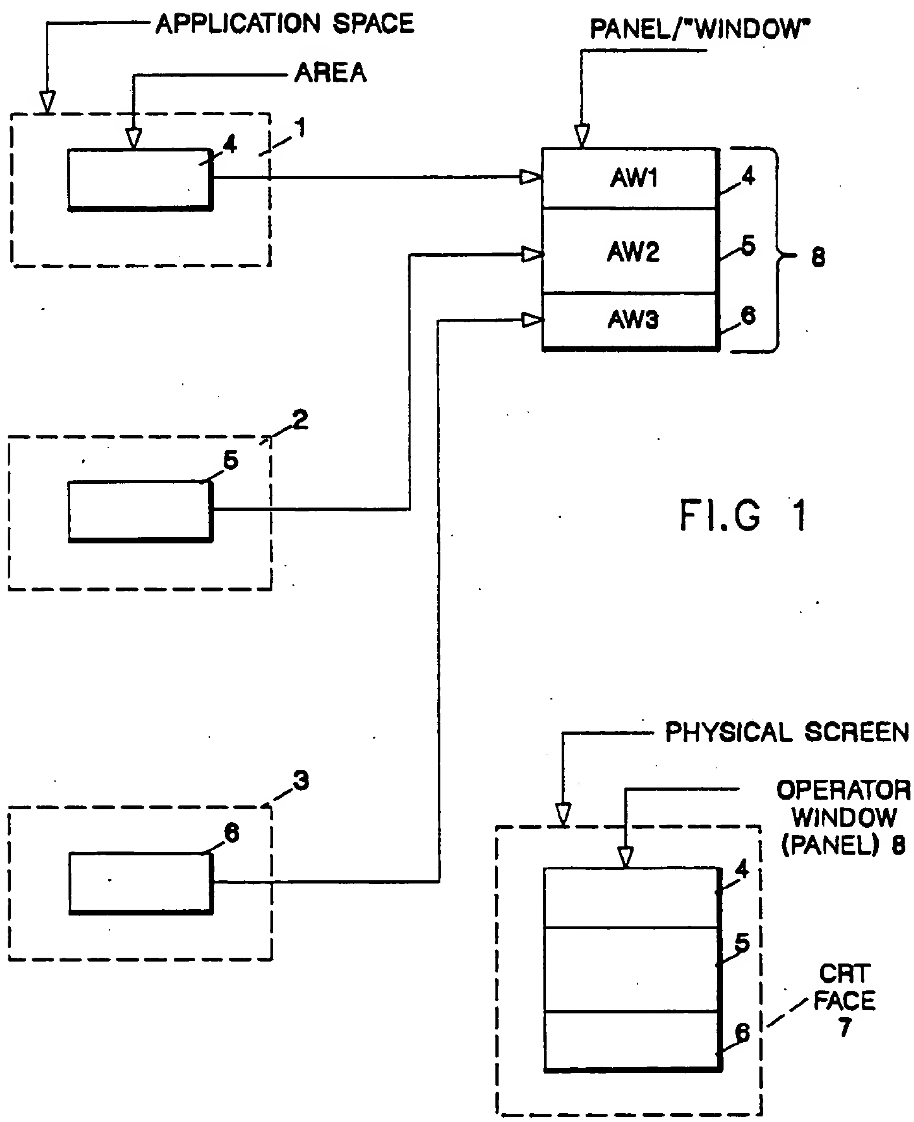
Primary Examiner—David L. Clark
Attorney, Agent, or Firm—Edward H. Duffield

[57] ABSTRACT

In display screen or system technology, a window is a viewing area on the video display. It may be the full screen region or a smaller region represented within a border of typically rectangular shape into which data from application programs and the like may be written for display. One or more windows may appear on the face of a video display screen. In the context of the present invention, the window areas are of variable size selected by the operator and resizing of the regions or areas within each variable window must be modified to suit the newly selected window size. Attributes associated with the regions to be placed within a given window include those for relative priority of display within the window, location within the window and the minimum dimensions of each region to be included within the window. Program controlled operations examine the minimum specifications for the regions to be displayed within a window in comparison with the operator-selected window size in which the regions are to be displayed, and aportion the available window space among the regions to be displayed in accordance with their relative priority and location in the window and their specified minimum sizes, and generate the control parameters necessary for recreating the window display with the appropriate regional spaces allocated and located within the window.

8 Claims, 8 Drawing Sheets




APPLICATION SPACE
AREA
PANEL/"WINDOW"
1
4
AW1
AW2
AW3
4
5
6
8
2
5
3
6
PHYSICAL SCREEN
OPERATOR WINDOW (PANEL) 8
4
5
6
CRT FACE 7

FI.G 1

FIG. 2

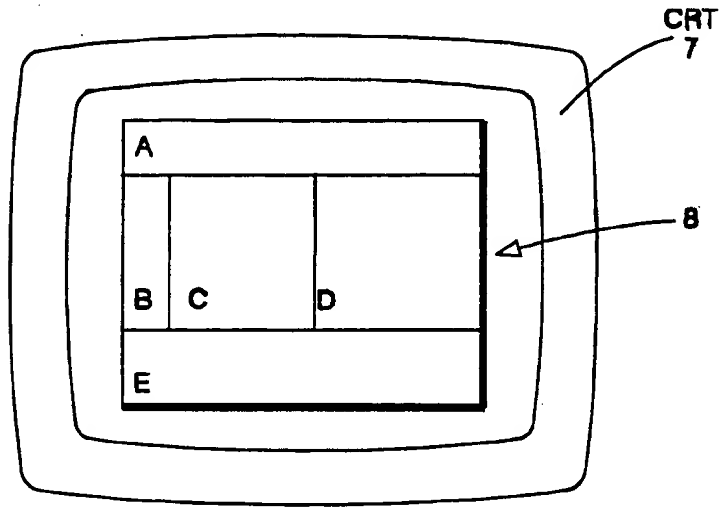


FIG. 3A

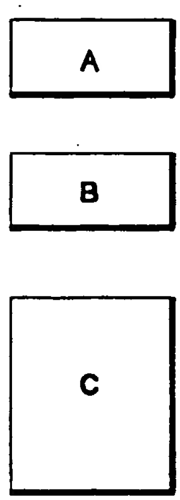


FIG. 3B

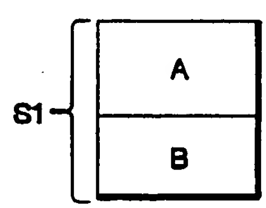


FIG. 3C

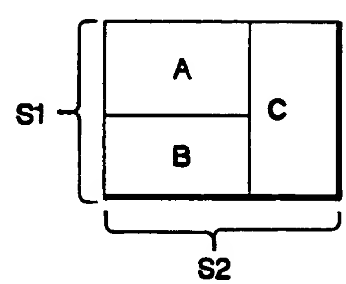


FIG. 4

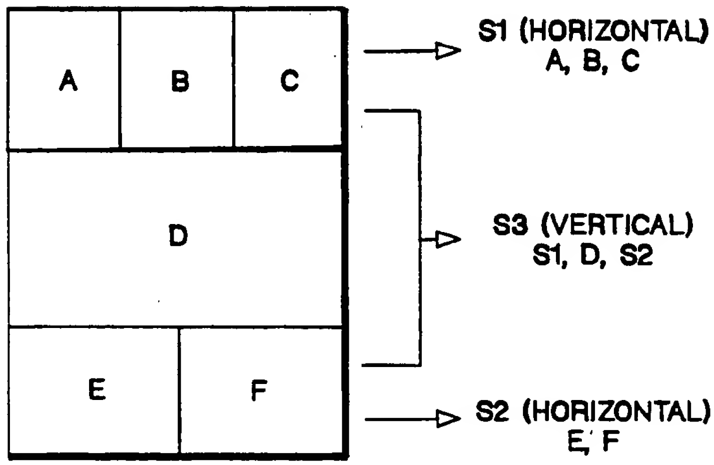


FIG. 5A

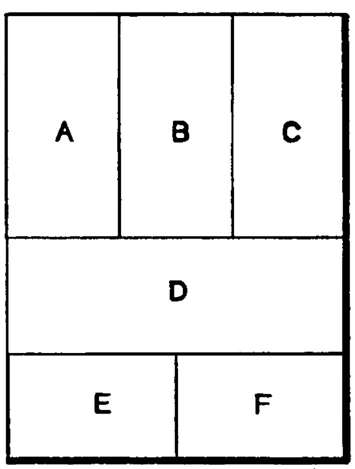
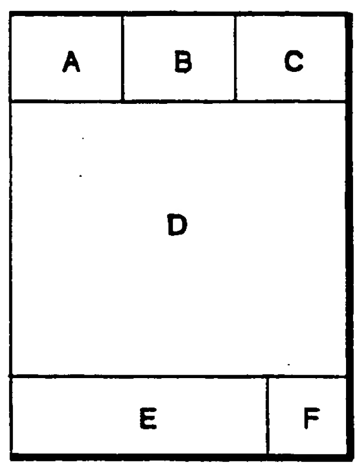


FIG. 5B



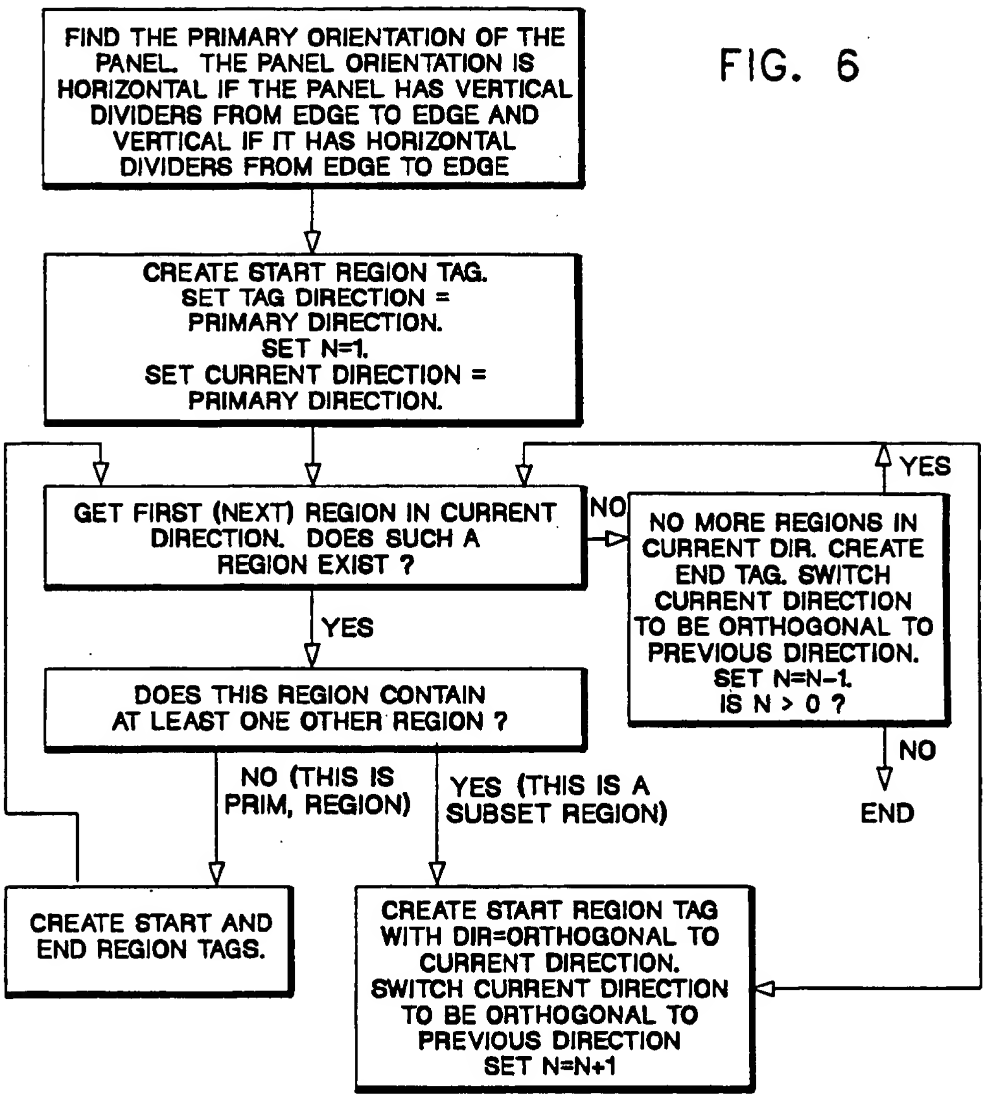

FIND THE PRIMARY ORIENTATION OF THE PANEL. THE PANEL ORIENTATION IS HORIZONTAL IF THE PANEL HAS VERTICAL DIVIDERS FROM EDGE TO EDGE AND VERTICAL IF IT HAS HORIZONTAL DIVIDERS FROM EDGE TO EDGE
CREATE START REGION TAG. SET TAG DIRECTION = PRIMARY DIRECTION. SET N=1. SET CURRENT DIRECTION = PRIMARY DIRECTION.
GET FIRST (NEXT) REGION IN CURRENT DIRECTION. DOES SUCH A REGION EXIST ?
NO
NO MORE REGIONS IN CURRENT DIR. CREATE END TAG. SWITCH CURRENT DIRECTION TO BE ORTHOGONAL TO PREVIOUS DIRECTION. SET N=N-1. IS N > 0 ?
YES
NO
END
YES
DOES THIS REGION CONTAIN AT LEAST ONE OTHER REGION ?
NO (THIS IS PRIM, REGION)
YES (THIS IS A SUBSET REGION)
CREATE START AND END REGION TAGS.
CREATE START REGION TAG WITH DIR=ORTHOGONAL TO CURRENT DIRECTION. SWITCH CURRENT DIRECTION TO BE ORTHOGONAL TO PREVIOUS DIRECTION SET N=N+1

FIG. 6

FIG. 7

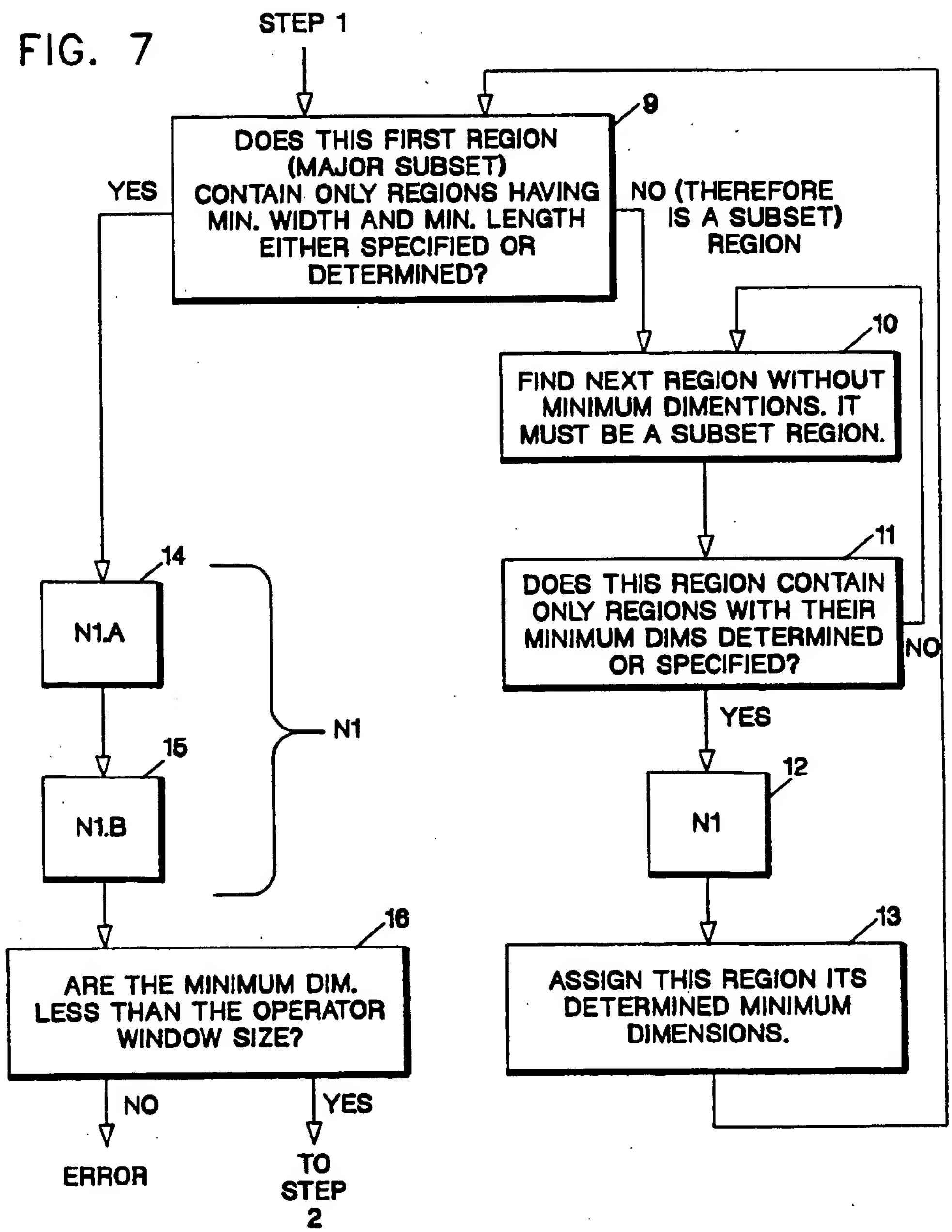
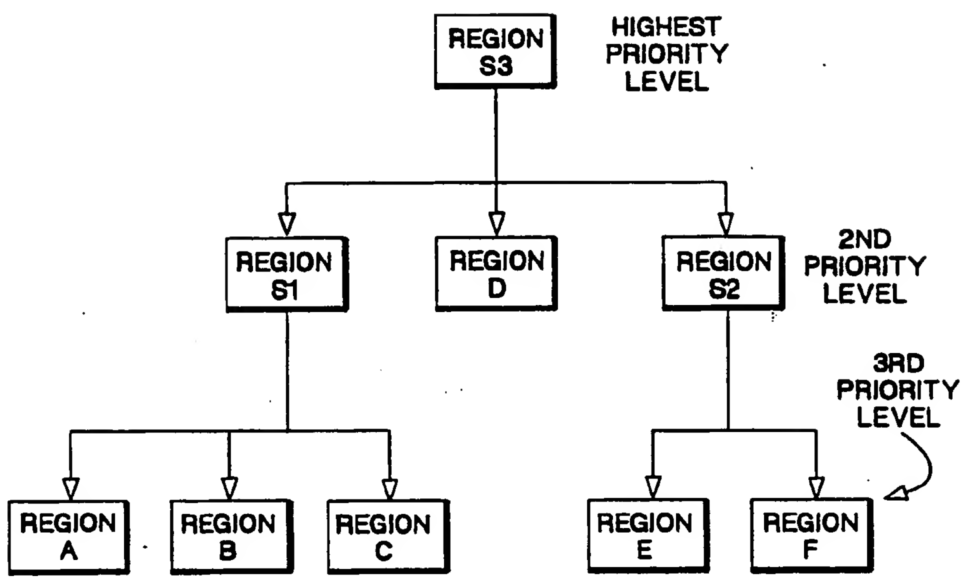

STEP 1
DOES THIS FIRST REGION (MAJOR SUBSET) CONTAIN ONLY REGIONS HAVING MIN. WIDTH AND MIN. LENGTH EITHER SPECIFIED OR DETERMINED?
9
YES
NO (THEREFORE IS A SUBSET) REGION
FIND NEXT REGION WITHOUT MINIMUM DIMENTIONS. IT MUST BE A SUBSET REGION.
10
DOES THIS REGION CONTAIN ONLY REGIONS WITH THEIR MINIMUM DIMS DETERMINED OR SPECIFIED?
11
NO
YES
N1
12
ASSIGN THIS REGION ITS DETERMINED MINIMUM DIMENSIONS.
13
N1.A
14
N1.B
15
N1
ARE THE MINIMUM DIM. LESS THAN THE OPERATOR WINDOW SIZE?
16
NO
YES
ERROR
TO STEP 2

FIG. 8


REGION S3
HIGHEST PRIORITY LEVEL
REGION S1
REGION D
REGION S2
2ND PRIORITY LEVEL
3RD PRIORITY LEVEL
REGION A
REGION B
REGION C
REGION E
REGION F

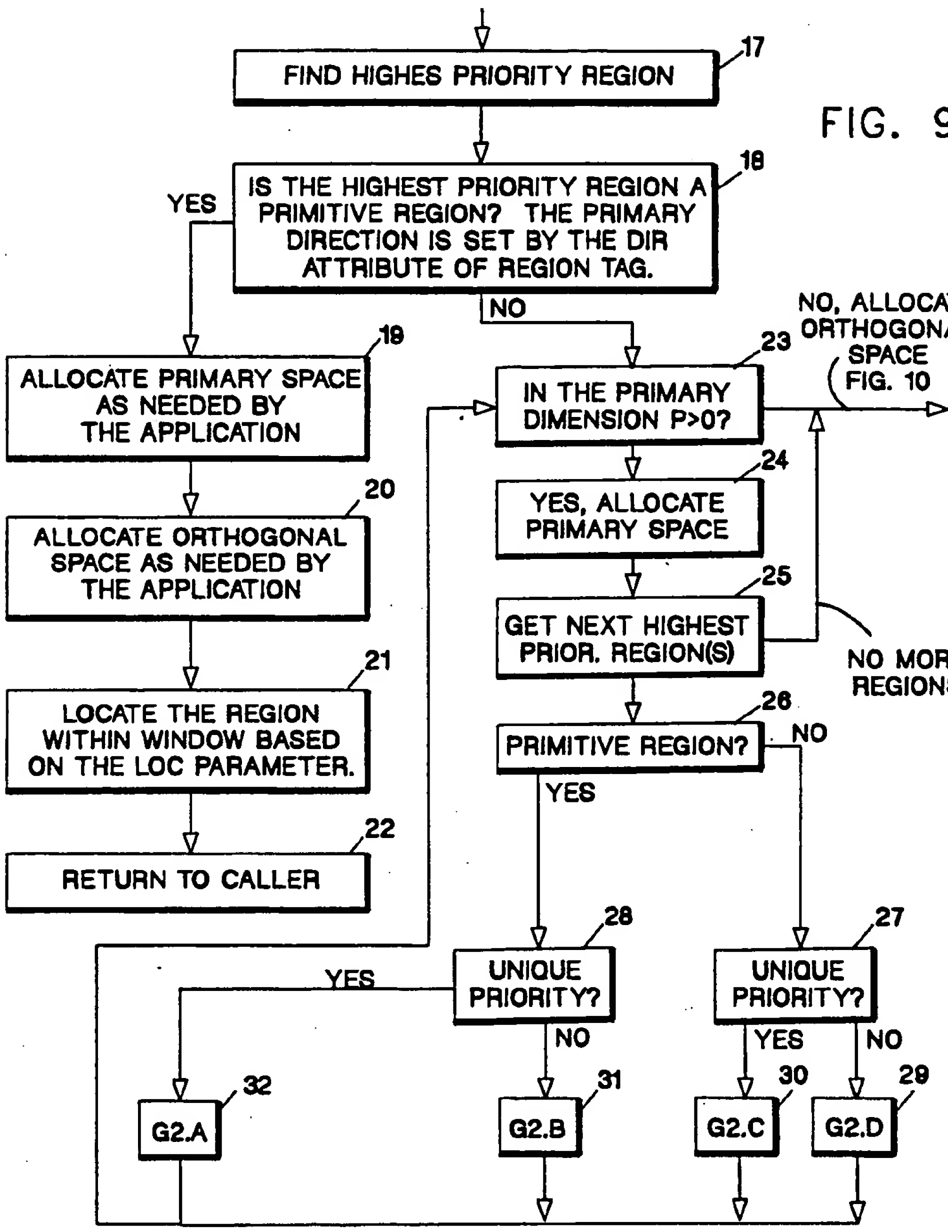

FIND HIGHES PRIORITY REGION
17
IS THE HIGHEST PRIORITY REGION A PRIMITIVE REGION? THE PRIMARY DIRECTION IS SET BY THE DIR ATTRIBUTE OF REGION TAG.
18
YES
NO
ALLOCATE PRIMARY SPACE AS NEEDED BY THE APPLICATION
19
ALLOCATE ORTHOGONAL SPACE AS NEEDED BY THE APPLICATION
20
LOCATE THE REGION WITHIN WINDOW BASED ON THE LOC PARAMETER.
21
RETURN TO CALLER
22
IN THE PRIMARY DIMENSION P>0?
23
NO, ALLOCATE ORTHOGONAL SPACE FIG. 10
YES, ALLOCATE PRIMARY SPACE
24
GET NEXT HIGHEST PRIOR. REGION(S)
25
NO MORE REGIONS
PRIMITIVE REGION?
26
NO
YES
UNIQUE PRIORITY?
28
UNIQUE PRIORITY?
27
YES
NO
YES
NO
G2.A
32
G2.B
31
G2.C
30
G2.D
29

FIG. 9

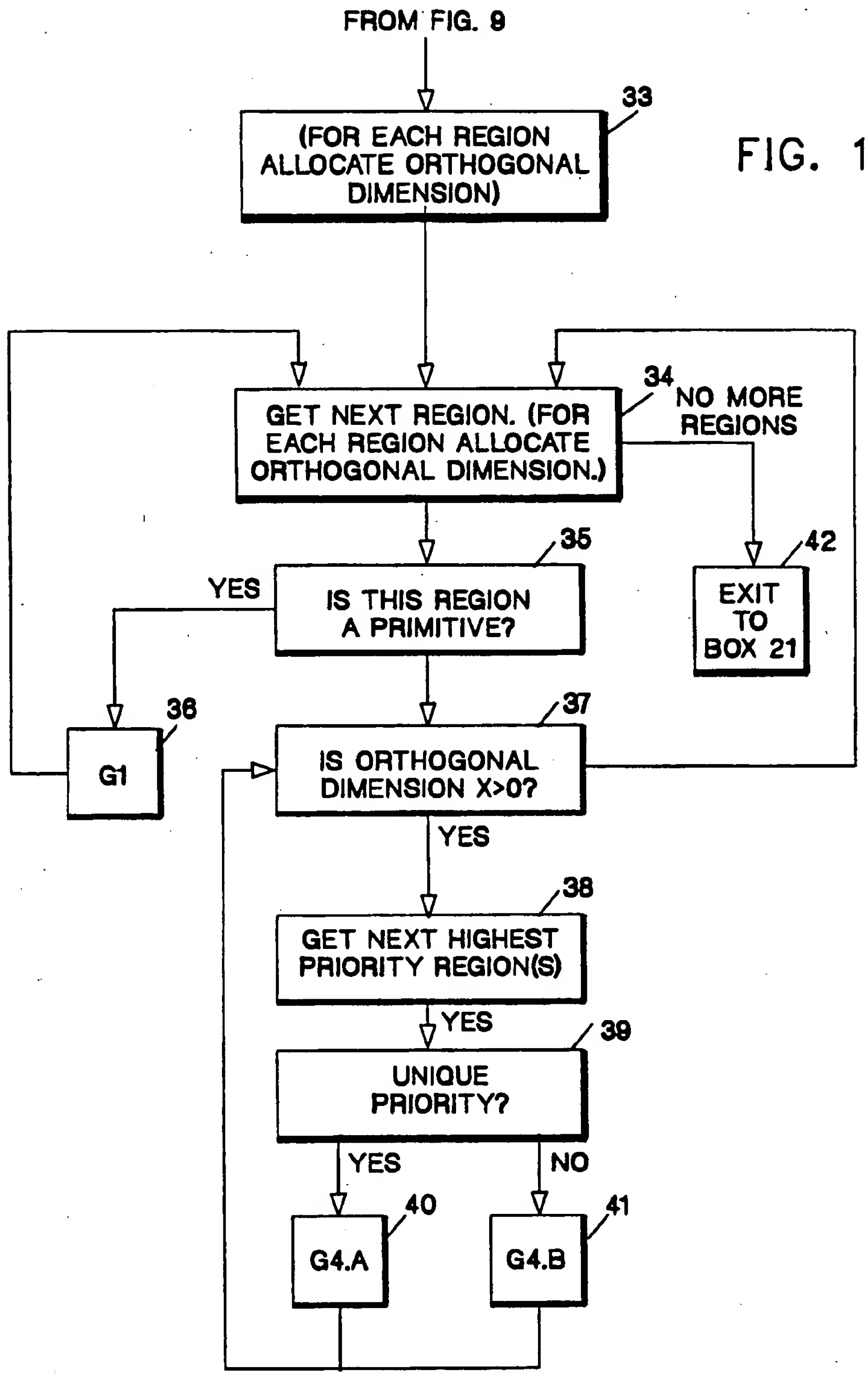

FROM FIG. 9
33
(FOR EACH REGION ALLOCATE ORTHOGONAL DIMENSION)
34
GET NEXT REGION. (FOR EACH REGION ALLOCATE ORTHOGONAL DIMENSION.)
NO MORE REGIONS
35
YES
IS THIS REGION A PRIMITIVE?
42
EXIT TO BOX 21
36
G1
37
IS ORTHOGONAL DIMENSION X>0?
YES
38
GET NEXT HIGHEST PRIORITY REGION(S)
YES
39
UNIQUE PRIORITY?
YES
NO
40
G4.A
41
G4.B

FIG. 10

SPACE ALLOCATION AND POSITIONING METHOD FOR SCREEN DISPLAY REGIONS IN A VARIABLE WINDOWING SYSTEM

FIELD OF THE INVENTION

This invention relates generally to computer systems having displays utilizing one or more data windows for manifesting or confining data within specified areas on the display screen. In particular, the invention relates to variable size windowing controls in which a collection of one or more regions for containing data are bounded by an operator window boundary of variable size.

PRIOR ART

A wide variety of prior patents exist in the general field of this invention. The apparatus and systems for controlling the display of data within fixed size windows on the face of a display screen or for controlling the formatting of data within regions or zones on the face of such a screen are fairly well known. For example, U.S. Pat. Nos. 4,598,384, 4,651,146, 4,653,020, 4,663,617, 4,698,779 and 4,731,606, all commonly assigned to the assignee of this application, may be cited. These patents show various details of systems, methods, and controls for the display of one or more data windows on the face of a display screen. For their teaching of apparatus and method for creating and displaying windows of data on a display screen, these patents are incorporated herein by reference. However, none of these references provides any means of automatically varying the size and shape of included regions within a window as the operator selects different sizes for the display window itself.

Other patents showing similar sorts of multiple window display apparatus and techniques which also fail to teach a method of varying automatically the sizes, shapes and locations of regions to be displayed within a window as the window size varies are U.S. Pat. Nos. 4,783,648 and 4,823,108.

In the foregoing patents, in order to change the size of a window, either the operator must specify the newly desired sizes, shapes and locations for each region or area within a window which is to be displayed in order to change the size of a window, or the windows are of fixed size only. Alternatively, if the size or location of the overall window frame is changed, the regions within the window are not varied but are exposed or occluded to a greater or lesser extent as the size of the window varies. None of the patents appears to offer a solution which varies the size of the regions or areas to be displayed within the window as the size of the window varies.

OBJECTS OF THE INVENTION

In view of the foregoing known difficulties with the prior art, it is an object of this invention to provide an improved display system for computer data in which one or more display windows may be of variable size and the regions or areas within the window are automatically expanded or reduced as the size of the window is accordingly increased or decreased.

It is the further object of this invention to provide an improved method of allocating display space within a window to the various regions or areas which are specified as required to be displayed within the window.

Yet another object of this invention is to provide an improved method of specifying regions for display within a window that facilitates automatic recalculation of the sizes and locations of regions as the size of display window is varied.

BRIEF SUMMARY OF THE INVENTION

In the preferred embodiment of this invention, unique algorithmic processes have been devised to utilize specified display region control indicators, together with specifications of the newly desired window size, in order to calculate new final dimensions and relative locations for the regions to be displayed within the window. Regional control parameters specifying the minimum dimensions in two mutually orthogonal directions are specified by the application whose data is to be displayed within a given region. Furthermore, an indicator for relative location within the window for the region and a priority value are both utilized (for establishing which regions may first receive extra available space, or, on the contrary, which regions may be truncated if the window size grows too small to accomodate all of the intended regions). An analytical process of first assessing whether the combined minimum sizes of the various regions to be displayed within a window exceed the total window dimensions is performed. If sufficient area exists within the window to allow at least the minimum specified areas for the regions to be contained within the window to be displayed, then the available space within the window is allocated according to a prescribed process of assigning a space available first in the primary direction of organization of the region (or grouping of regions) to be displayed within the window and then in the direction orthogonal thereto, with the spaced assigned to each member region within the window on the basis of its priority and minimum size specifications in general. Special processes for assigning the available space to primitive display regions, i.e. ordinary rectangles, or to subsets of display regions, i.e. groups of two or more rectangles, to occupy the space within a defined window frame are developed and explained.

The foregoing objects of the invention and still others not specifically enumerated are met in a preferred embodiment thereof as will be described in greater detail with reference to the drawings in which:

BRIEF DESCRIPTION OF DRAWINGS

FIG. 1 illustrates the general concepts of display windows, areas and subsets of areas to be displayed within the windows.

FIG. 2 shows a typical CRT screen with a single window having five sub areas or regions within it forming a subset of regions for display.

FIG. 3 comprises FIGS. 3A, 3B and 3C and illustrates the concepts of primitive regions, simple subset regions and complex subset regions, respectively.

FIG. 4 illustrates the concepts of complex subset regions for display within a window and introduces the concepts of vertical and horizontal orientations for the subsets.

FIG. 5 consisting of FIGS. 5A and 5B illustrates the difference in appearance that can occur in two complex subset window displays having the same number of and arrangement of regions within them, but having regions of differing priorities specified which have been processed by the method of the preferred embodiment.

FIG. 6 is the analytical flowchart for the process of analyzing a region or group of regions specified for

display within a window for generating a specification for the complete composite regional display.

FIG. 7 illustrates the flowchart for the process of calculating the minimum length and width of the composite or subset of regions to be displayed within a window.

FIG. 8 illustrates schematically the concept of regional priorities of regions for display within a window.

FIG. 9 illustrates the process flows for allocating space within the window in the primary direction of orientation of the subset of regions to be displayed.

FIG. 10 illustrates the process flowchart portion which relates to the allocation of display space in the direction orthogonal to the primary direction of orientation of the regions for display within the window.

DESCRIPTION OF PREFERRED EMBODIMENT OF INVENTION

As alluded to earlier, the phrase "variable windowing" in this invention is used to refer to a representational model in which a collection of one or more regions in which information is to be displayed is bounded by an operator window frame of variable size. The size or shape of the operator window frame is changed by a computer or terminal operator who desires to relocate a window on the face of a display screen or to change its size or shape. If the size or shape of the window frame is changed, the contents of the window in the normal prior art systems will be truncated if the window frame is reduced or will be enlarged if the frame size is increased. The effect of this in prior art systems is to display more or less of the data within each region or area within the window. This is undesirable, since as the window is shrunk, so much information may be truncated that the regions or areas within the window become essentially meaningless. Similarly, if the size of the window frame is increased, but the amount of data to be displayed is relatively small or limited, increasing the frame size causes a lot of "white space" displayed around the information within each region which tends to make finding the information within each region somewhat awkward and inconvenient.

It would be most desirable if the information display regions making up a window were expanded or contracted somewhat proportionally to the expansion or contraction of the window itself, keeping at least a minimum required amount of display space available and adjusting the maximum to a degree that is suitable for the amount of information to be displayed. Such a complex process of assigning and reassigning of available space to the one or more display regions within a window may be easily accomplished by a programmer or operator changing the parameters that define each area, but this is a cumbersome process, especially when multiple regions or areas are exhibited within each window.

The present invention solves this problem by providing automatic processing methods which are driven by the operator's selection of a new window frame size. These processing methods operate utilizing specified minimum parameters, location indicators and area priorities and the analysis of the direction of primary orientation of any subsets of areas within the display. The processing methods recalculate resulting region sizes to fully utilize all of the space in the redefined window frame size. This approach also solves another problem: that of specifying from one program, which contains the parameters for constructing a given window display with its inherent regions, all that is necessary to reconstruct a similar display in another program or system which may have a differently-specified total window frame size or display area availability.

Succinctly stated, the challenge of this invention is in finding a suitable method of implementation by which a programming interface can handle the generalized information about a specific layout of a screen display and also manage a complex set of rules necessary for adapting the display to new window sizes as the size of the window changes. For example, if program A were a program that created and recreated displays based on the size of the operator window that is specified, and if program B were a specific application of program A, how would program B indicate to program A the desired structure for building the overall screen display that program B wants?

FIG. 2 illustrates this problem somewhat graphically. In FIG. 2, a CRT screen face is depicted having a single window 8 comprising a collection of regions A through E that are presumed to have been specified by the application program or programmer. What must the application program or programmer communicate to the display program in order to reconstruct this display? First, the information that region A is always on the top, region E is always on the bottom, and regions B, C and D are in between the regions A and E. Furthermore, region B is located to the left of region C and C to the left of region D. When the window is made larger in the vertical direction, regions B, C and D should expand. However, region A should never be more than one line deep, and region E should never be more than two lines deep in this arbitrary example. If the window is made smaller in the vertical direction, the system should make sure that region A is always visible, i.e. has the highest priority, should make certain that regions B, C and D are always at least three lines deep and that region E has the lowest priority and may be truncated first followed by truncation of any additional space beyond the minimum three lines in regions B, C and D. When the width of the operator window changes, the system should make sure that region B never gets wider than a single column width and that regions C and D should expand or contract equally as the operator width changes.

This is not an exhaustive list of all of the specifications that would be necessary to communicate to a display program from an application the rules to be followed in changing the allocation of space as the window is increased or decreased, but it illustrates the problem very graphically. In fact, none of the prior art approaches address this problem at all, presumably because of the complexity of solving it.

The present invention provides a means for managing the area for display within a window with a program that operates utilizing standard screen display definition languages. Such languages are those based on international standard (ISO) markup languages that allow programs or programmers to communicate with programs. These are defined in a general fashion and specific examples will be given later herein. The definitions of screen displays utilize tag language which are sets of predefined commands for specifying minimum size, primary direction of orientation of the region or subset, the subset's relative location in the window and the subset's priority by means of various indicators. Indicators for the start of the area definition, the relative size, location and priority of the areas within the window may all be specified.

Returning to FIG. 1, a variety of application programs are illustrated as the schematic boxes 1, 2 and 3. Each of these programs may be presumed to have some function that results in data being created that would fill a given area identified as areas 4, 5 and 6. It is further presumed that the system operator wishes to display a screen or window having a subset 8 made up of an arrangement of the areas 4, 5 and 6 from the several application programs 1, 2 and 3 as shown. The CRT face 7 will contain the operator window frame 8 and the various areas 4, 5 and 6 arranged in a subset. This subset will later be seen to be a "vertical subset".

In FIG. 2, it will be seen that a screen display or "window" will be made up of non-overlapping regions or areas that may always be described in terms of rectangles or collections of rectangles. In this application, a single rectangle is referred to as a "primitive" area, and a rectangular collection of such rectangles is referred to as a "subset" area. A subset contains one or more regions and the regions themselves can be either primitive regions or further subsets. FIG. 3A shows three primitive regions A, B and C, while FIG. 3B illustrates a subset region, S1, containing two regions A and B, both of which are themselves primitive regions. FIG. 3C illustrates a complex subset, S2, having two general regions, one of which is the simple subset S1 having regions A and B and the other of which is a primitive region C; region S1 and region C are grouped together horizontally in FIG. 3C.

The foregoing raises the notion of the primary direction of orientation of a subset. Within the realm of display screen technology, the common directions of orientation are vertical and horizontal. There are, accordingly, vertical and horizontal subsets of regions. A vertical subset is one in which the regions or areas for display within the window are arranged vertically from top to bottom in the window. A horizontal subset contains regions that are arranged horizontally from left to right. In FIG. 3B, subset S1 is a vertical subset because regions A and B are arranged one over the other. This might be easily found from analysis by discovering that a divider, i.e. the line between regions A and B extends from border to border within the window in a horizontal direction. That is, a horizontal divider conotes a vertical subset and a vertical divider, as a corollary to this notion, conotes a horizontal subset. The subset S2 in FIG. 3C is a horizontal subset composed of the regions subset S1 and primitive region C.

A more complex window display is illustrated in FIG. 4.

In FIG. 4, the overall window display (the outer box or frame within which all of the rectangles are contained) contains three subsets and six primitive regions. The primitive regions are lettered A through F and the subsets are as follows. Subset S1, a horizontal subset, consists of regions A through C. Subset S2, another horizontal subset, consists of regions E and F. Subset S3, a vertical subset, comprises subsets S1, primitive region D and subset S2.

In FIG. 4 the major or definitive subset is that subset which describes the entire screen or window display. In FIG. 4 subset S3 contains the definition of all of the regions and subsets that make up this hypothetical display window. As the operator selects a different size for the outermost rectangle or frame within which all of the primitive regions are contained, the size of the major subset S3 would vary as a function of the window size. It would be necessary to either specify precisely what the redistribution of space should be amongst the members A through F or to provide some automatic technique for recalculating the sizes to be displayed. This is done in the present invention.

The person who originally specifies the appearance of the screen display within a given window (called a panel) would describe an example (panel) as shown in FIG. 4 using panel and region tag statements as shown in Table 1 and Table 2 below. The panel definition prescribed by such a programmer is begun utilizing a panel tag and is closed utilizing a matching end panel tag. The panel tag has command identifiers that establish the panel name, the identity of the help text that will pertain to the panel display as a whole, the overall panel, i.e. window dimensions, the number of message lines to appear on the panel, cursor placement control indicators and a panel title as well as the usual tags for defining areas, instructions at the bottom, dividers, data columns, data fields, information, etc.

TABLE 1

```
Panel Tag ____________________
<PANEL - start of Panel Definition
      NAME = panel-name
      [ HELP = help-panel-name ]
      [ DEPTH = initial-depth-value ]
      [ WIDTH = initial-width-value ]
      [ MSGLINES = 0 | nbr-msg-lines ]
      [ KEYLIST = key-list-name ]
      [ CSRAREA = area-identifier ]
      [ CSRFIELD = field-identifier ]
      [ CSRINDEX = index-value ]
      [ CSRPOS = position-value ]>
[ panel-title-text ]
.
.- AB          tag
.- AREA        tags
.- BOTINST     tags
.- DIVIDER     tags
.- DTACOL      tags
.- DTAFLD      tags
.- INFO        tags
.- LSTFLD      tags
.- REGION      tags
.- SELFLD      tags
.- TOPINST     tags
.- UC          tags
.
</PANEL> - End of Panel Definition
```

In Table 1 above, "<PANEL" indicates the beginning of the panel definition. The end of the panel definition is indicated by the matching end tag "</PANEL>" as shown in Table 1. The name is the panel name and is a required field. It contains the name of this panel of display information. The name used as the panel identifier can be displayed as an end user option.

The help portion is optional and is the name of the help text panel that is defined with the help tag. It identifies help text that pertains to the panel as a whole and is stored in the commonly accessible area accessible by the application program. It is displayed when the operator requests help and the cursor is not otherwise on a panel element that has its own help text specified for it. Depth and width are attributes specifying the initial depth and width of the window being defined. Once a window is established, the end user can resize it. The "message" line tags an attribute that specifies the number of message lines that are to be reserved on this panel display. "Key list" is an attibute which specifies the name of a key list for the operator's keys that are associated with this particular panel of display information. The "cursor area attribute", together with "cursor

field", "cursor index" and "cursor position", are used to control the placement of the cursor on the display whenever this specified panel of information is displayed. These attributes specify the identifier for the area tag that identifies where the cursor should initially be located whenever the panel is displayed. The panel title text is optional and specifies the title that will appear on the panel when it is displayed.

In this preferred embodiment, the programmer is also required to specify the regions that will appear within the window utilizing the tag language as shown in Table 2. The purpose of the region tags is to specify space within the panel definition within which output from other tags is to reside, i.e. the subareas within which data is displayed within a given window. The parameters in the region tag are used to specify information about each region and the way the space within the region is to be allocated. The region tag begins the region and is used to separate parts of a screen or panel definition from other regions. It is also used to control the panel layout in the methods which will be described later. A region may be started at any point within a panel definition and may also start within an earlier defined region, i.e. it may be nested within a previous region.

Table 2 shows an example of region tag.

TABLE 2

```
Region Tag
<REGION
    [ NAME = region-name ]
    [ MIN = row,column ]
    [ MAX = row,column ]
    [ DIR = VERT | HORZ ]
    [ LOC = TOP | BOT, LEFT | RIGHT, CENTER ]
    [ PRIORITY = priority ]>
.
.- (All tags allowed within panels)
.
</REGION>
```

As shown in Table 2 above, the region tag, "<REGION", indicates the beginning of a region definition. A matching end tag, "</REGION>", ends the definition for a region. Within the region tag, NAME gives the region name used when the application programmer wishes to position a message or cursor within a given region that is being specified. MAX is the maximum number of rows or columns to be allocated to a given region, and MIN is the minimum number of rows and columns required for the region. The minimum and maximum parameters are really only valid on primitive regions, i.e. those that do not contain any other regions. The "Direction" parameter tells the compiler operating the process (that will be described later) which direction is the "primary" direction of orientation for the overall window as it is subdivided into other regions. The default value is "vertical", it will cause a vertical list of panel regions to be compiled. The "location" parameter specifies how the region will be placed in a subset relative to other regions in the same subset within a window. "Top" and "bottom" are valid for vertical subsets and "left" and "right" are valid for horizontal subsets. A "center" definition is also possible and is valid for both horizontal or vertical subsets. The default values are: "top" and "left". Finally, the "priority" parameter is utilized to specify which region, when two or more regions within a window have an indeterminate dimension along the primary axis, is to be allocated space preferentially. The priority of allocation is controlled by the priority parameter. All regions having equal priority receive space in equal amounts. Regions of differing priority receive space according to their relative priorities, with the higher number priority receiving extra space sooner than those with lower number priorities. The default priority value is 0, and the maximum is, arbitrarily, 10.

Table 3 illustrates a completed panel definition, i.e. a "window definition" for constructing the display as shown in FIG. 4.

TABLE 3

<panel name=example> Example Panel	
<region>	/*This first region tag defines the start of (major subset) region S3
<region dir=horz>	/*This second region tag defines start of S1
<region min=10,5>	/*This third region tag defines start of region A
</region>	/*This ends region A
<region min=5,8>	/*This fourth region tag defines start of region B
</region>	/*This ends region B
<region min=12,6>	/*This fifth region tag defines start of region C
</region>	/*This ends region C
</region>	/*This ends subset S1
<region min=9,25>	/*This sixth region tag defines start of region D
</region>	/*This ends region D
<region dir=horz>	/*This seventh region tag defines start of region (subset) S2
<region min=5,25>	/*This eighth region tag defines start of region E
</region>	/*This ends region E
<region min=7/30>	/*This ninth region tag defines start of region F
</region>	/*This ends region F
</region>	/*This ends subset S2
</region>	/*This ends (major) subset region S3

Table 3 is self explanatory and shows the completed specification parameter definition for constructing a display within a window as shown in FIG. 4. If a program B, for example, were describing this overall screen display to a program A which would display the specified regions within a window that it had available, then these would be the specified parameters. The definitions in Table 3, together with the processes that will be described later, are all that is necessary to reconstruct the display in FIG. 4 in a window of any given size. It will be noted that in Table 3, the sizes for the primitive regions are not indicated. These must be determined by the controlling program A utilizing the methods as described later when the size of the selected operator window frame is known. Instead, program B only indicates the arrangement, minimum sizes and relative priorities of the primitive regions within the composite window. Program A will create the overall window display such as shown in FIG. 4 based on the size of the selected operator window and on any information provided by program B with the tags as shown in Table 3.

A striking example of the difference that specification of minimum sizes and priorities can make is seen in FIGS. 5A and 5B in which the window display of FIG. 4 is recreated with two different appearances that result when differing priorities and minimum sizes are specified. The concepts of horizontal and vertical subsets along with the information about relative location of areas, their priorities and minimum sizes are all that are necessary, together with the method which will be described below, to reconstruct or, as it is used herein,

position and allocate the regions to be displayed within a window of any variable size selected by an operator.

In Table 3 above, the order in which the regions are defined determines their arrangement within subsets. For example, when defining subset S1 with the primary direction "horizontal" as shown in Table 3, if region A is defined first with regions B and C defined second and third, this will indicate that the regions should be arranged with A to the left of B, B in the center and C to the right of B. In priority order, each member will be given its minimum amount of specified space, if possible. After that, space will be allocated to each region based on its relative priority compared to the others within the window. The "minimum" space could be a conditional minimum in which there would be no error condition if there were not enough space to fill all of the minimum requirements. In such an event, regions with the lowest priority would simply be truncated, or might disappear altogether, if the minimum space required is not available in the newly specified window frame size.

It may be apparent that an analysis of any specified window of regions can be carried out to find rectangular regions of application data that are to be treated uniquely when resolving the overall window definition to a new window size. The process is illustrated in FIG. 6 in a flowchart. The process begins with the largest region possible that defines the entire window array and then examines the array for the next largest orthogonal set of regions contained within it, if any. The next largest set of regions are then distinguished by having either a horizontal or vertical divider that extends from window boundary to window boundary. The process of finding orthogonal sets of regions within regions continues until there are no more sets of regions. Utilizing the process shown in the flowchart in FIG. 6, any specified window display consisting of one or more regions can be analyzed to generate the definition list for the entire window display as shown in Table 3.

Once a window display has been described utilizing the panel and region tags as noted above, a receiving program can create a panel to fit any size specified operator window. The window display is rebuilt or "resolved" each time the operator window size is changed. The size of each contained region or area will be based upon the minimum specified size thereof and its relative priority as indicated in the tags. The following algorithms are used for recreating, creating, i.e. "resolving", the new window displays in response to the input of the tag specifications and the minimum window size selected by an operator.

The first step as shown in FIG. 7 is to determine whether the chosen operator window size is large enough to accomodate the full array of specified regions. The process is as follows:

N1. For the major subset, determine the minimum subset dimensions as follows in order to determine if the panel will fit within the given operator window*:

N1.*a*. Determine the minimum subset orthogonal dimension by finding the largest of the minimum orthogonal dimensions of all of the regions.

N1.*b*. Determine a minimum subset primary dimension by adding together the minimum primary dimensions of all of the regions in the subset.

If a subset contains regions that are themselves subsets, the minimum dimensions of each such region must be determined first. If the regions are primitive regions, the minimum dimensions of these regions will be defined in the tags such as in Table 3. The flowchart in FIG. 7 shows the method of determining the minimum dimensions to determine whether the panel, i.e. window array specified, will fit within a given operator window size that has been selected by the operator. In the flowchart of FIG. 7, "N1.*a*" and "N1.*b*" and "N1" refer to the steps in the algorithm above.

The flowchart in FIG. 7 begins in box 9 and eventually ends in box 16 with a determination that the minimum dimensions either are, or are not, less than the specified operator window size. If the minimum dimensions of the specified window display are larger than the specified available window size selected by the operator, an error condition can be indicated or, if desired, the default condition can be to display, i.e. "resolve" the overall subset with the lowest priority members truncated entirely. However, assuming that the minimum dimensions of the specified operator window size are larger than or equal to the minimum size necessary for the total array as found from the process in FIG. 7, step 2 of the process of resolving each subset to create the new display is begun.

Step 2 begins with finding the major subset, i.e. the one which defines the overall array of regions making up a window, and then resolves each subset and resolves each region that is itself a subset. In this context, to "resolve a subset" means to determine the final dimensions of the subset of regions, the final dimensions of each region within the subset, and the arrangement of the regions and any "white space" left over within the selected window size. In resolving a subset, the maximum potential window dimensions are utilized and any difference between the maximum available window dimensions and the final dimensions becomes the "white space" in the final window display which is allocated in accordance with the priority and location parameters. The maximum potential dimensions of the major subset defining any given window display are the available length and width of the operator-specified operator window frame size.

FIG. 8 illustrates the hiearchical ordering of regions within a given window display such as that illustrated initially in FIG. 4. The highest priority level for resolution is the subset S3 that contains in it the definition of the entire window array. FIG. 8 illustrates this concept in which the highest priority level contains only region S3. The next echelon contains regions S1, primitive region D and region S2. These are all of equal priority level and are resolved second. Finally, regions A, B, C, E and F are at the third priority level and are resolved last. The hierarchical priority levels are utilized for assigning space, since priorities specified for a given region are only compared with other regions at their same level in the hierarchy, i.e. a priority 10 region F would not be compared with a priority 10 region S3, but only with any equal-level priorities specified for members A, B, C or E in FIG. 8. This will be understood in greater detail when the flowchart for the resolution pricess in FIGS. 9 and 10 is discussed.

For the step of resolving the areas in a subset the process is as follows:

G1. For each region R that is a primitive region, set the final orthogonal region of R to be the smaller of:

1. the potential orthogonal dimension of the subset in which the region lies or

2. the orthogonal dimension of the application space associated with R.

Step G2.

Divide the potential primary dimension, i.e. the maximum window dimension of the subset between the regions within the subset as follows: allocate the primary dimension P to each region in the order of priority of the regions and in an amount to their minimum specified primary dimension; next allocate the orthogonal dimension X to each region in the subset in the order of priority among the regions making up the subset according to their specified primary orthogonal dimensions.

Let capital P represent the primary axis dimension to be allocated in the display. Initially P will be the maximum window dimension in the primary direction.

If capital P is still greater than 0 after all regions have had their specified minimum primary dimensions allocated to them, this means that there is still some primary dimension within the window to be allocated. This space is then allocated by allocating additional primary allocations to each region within the subset and decreasing capital P by that amount allocated. Allocation is begun with the highest priority region within the subset until its maximum allocation as specified has been achieved or P is exhausted and then moving on to the next highest priority region, if any, until all regions have been processed or the remainder P becomes 0.

The rules for allocating primary dimensions within the window to a region within the subset are as follows:

Step G2.*a*

If the region is a primitive region then, if no other regions have the same or higher priority, in addition to what has already been allocated as the minimum, allocate to that region either all of the remaining P or that portion of P which makes the total amount allocated equal to the primary dimension of the associated application space. Subtract the amount allocated from the remainder P and set the final primary dimension of the region to whatever has been allocated. Application space is that space needed by the data within the region and may be identified from its application program.

If there are other regions within the subset that have the same priority, then divide P by the number of regions having equal priority and call the result P'. In addition to what has been allocated for the minimum dimensions for each such member, allocate to each of the regions either all of P' or that portion thereof which makes the total amount allocated equal to the primary dimensions associated with these regions in their application of space. The final primary dimension of each region will be set to be equal to the total amount allocated. If the region was not given all of P', then P' is recomputed for any other regions that have equal priority and processed in the same way until all of P' has been exhausted.

If the region is a subset itself, it is axiomatic that the axis of orientation or organization of the subset must be orthogonal to the primary specified direction.

Step G2.*c*

If no other subsets in this orthogonal direction have equal priority, for each region in this subset in addition to what may have already been allocated for the minimum primary dimensions, allocate to that region the smaller of either the remaining primary dimension P or the primary dimension of the application space associated with that region. Set the final primary dimension of this subset to be equal to the primary dimension of the largest region within the subset. Subtract from P the amount finally allocated for this final primary dimension.

Step G2.*d*

If there are other subsets with the same priority in this orthogonal direction, divide P by the number of subsets having equal priority calling the result P'. For each of these subsets add to the minimum primary dimension of each region in the subset the smaller of either the primary dimension of P' or the primary dimension of the application space associated with that region. Set the final primary dimension of this subset to be equal to the primary dimension of the largest region, that is, the dimension in the primary direction for the subset. Subtract from P' any additional amount allocated for this final primary dimension. If the subset has not been allocated all of P', recompute P' for the remaining subsets having any equal priority and if there are none to those having next lowest priority, etc. Process each region with equal priority in the same way. If the last subset has not been given all of P', the amount left over will be assigned back to primary space P to be allocated as follows.

Step G3

If there is any difference between the potential maximum, i.e. the specified window dimension, and the final dimension that is allocated in the subset, arrange the regions and any remaining space in the direction P, i.e. the white space, based upon the location parameter for the regions making up the subset.

Step G4

Finally, it is necessary to divide the potential orthogonal maximum dimensions of the subset between the regions as follows: Let X represent the orthogonal dimension still to be allocated to the regions within the subset after all have been given their minimum orthogonal dimensions. If X is still greater than 0, then allocate the orthogonal dimension to each region and decrease X by the amount allocated beginning with the highest priority region and moving on to the next lower priority until all regions have been processed or the remainder of X is 0. In order to allocate orthogonal dimension to a region which is a primitive region:

Step G4.*a*

If no other regions have the same priority, then in addition to what has already been allocated for the minimum orthogonal dimension, allocate for that region either all of the remaining X or that portion of X which makes the total amount allocated equal to the orthogonal dimension of the associated application space, subtracting the amount allocated from X. Set the final orthogonal dimension of this region to be the total amount allocated.

Step G4.*b*

If there are other regions with the same priority, divide X by the number of such regions and call the result X'. In addition to what has already been allocated for the minimum orthogonal dimension, allocate to the region either all of X' or that portion thereof which makes the total amount allocated equal to the orthogonal dimension of the associated application space. Set the final orthogonal dimension of the region to the total amount allocated. If this region has not exhausted all of X', then recompute X' for any remaining regions having equal priority and process each region with equal priority in the same way.

FIG. 9 illustrates the process of this step 2 of allocation in a detailed flowchart. Beginning in box 17, the first step is to find the highest priority region, i.e. in this context this means to find the region such as in FIG. 8 which hierarchically has the highest order, i.e. the one which is the major subset specifying the entire contents

of the desired window display. The process continues in box 18 where the highest priority region is checked to determine whether it is a primitive region. The primary direction is found from the direction attribute in the region tags and it is the primary direction space which is allocated first. Assuming that the highest priority region in box 17 is found in box 18 to be a primary region, the flow goes to boxes 19 and 20 where the primary space, i.e. the space in the primary axis of orientation, is allocated as needed and then the orthogonal space, i.e. the dimension at 90 degrees to the specified primary direction of orientation is allocated. Next, the regions are located within the window based on their location parameters which automatically results in placing the white space relative to the specified location for the regions, and the process is exited in box 22. However, assuming that the highest priority region found in box 18 is not a primitive region, the process continues to box 23 through 32 until finally there are no more regions to be allocated any space in the primary dimension. FIG. 9 is then exited from box 23 to the process of allocating the orthogonal space as shown in more detail in FIG. 10.

In FIG. 10, the process is begun in box 33 for computing the allocation of the orthogonal dimension for each region. It continues to box 34 where the next region to be processed is fetched, to box 35 where the region is examined for being primitive or not and continues through box 40 or 41 until all of the space has been assigned and the regions are exhausted in which case the system shown in this process exits through box 42 back to box 21 in FIG. 9 to locate the regions within the window based upon their location parameters.

In each of these flowcharts 9 and 10, the references within the boxes to steps "G2*a*", "G2*b*", etc refer to the overall description of the algorithm given above.

Having therefore described our invention with reference to a preferred embodiment thereof, it will be apparent to those of skill in the art that numerous departures from the specific algorithms given may be made without departing from the generic process for analyzing the specified window display and recreating similar displays within windows of various sizes after recomputing the allocation of space in the primary and orthogonal directions.

Therefore, what is contained in the following claims is intended by way of example only and not of limitation in which what is claimed is:

1. A computer-implemented method of controlling construction of visual window displays from area specifications describing relative area positions, priorities and minimum sizes comprising the steps of:

determining whether specified minimum dimensions of a composite of specified areas to be displayed within a window are equal to or less than available space within the window; and

if sufficient space is available within the window, allocating final dimensions and arrangement from the available space by first allocating dimensions in a primary axis of orientation and then allocating the dimensions in an axis orthogonal to said primary axis.

2. Method as described in claim 1 further comprising steps of:

said allocating of available window areas among said specified regions is determined by the specified minimum size, priority and position.

3. A method as described in claim 2 further comprising:

allocating the available space of said window among regions to be displayed therein by assigning window space in the primary axis of orientation of the composite regions to be displayed to said regions in the order of their priorities as indicated by said priority specifications.

4. A method as described in claim 3 further comprising:

arranging said regions within said window space in accordance with the relative area position specifications.

5. A computer-implemented method of generating a display of regions within a viewing window comprising steps of:

encoding region control indicators for controlling the generation of region displays, said indicators comprising a primary axis of orientation indicator, a relative regional location indicator, a priority indicator and a minimum dimension indicator; and

allocating space within a specified viewing window among said specified regions by utilizing said indicators in comparison with corresponding characteristics of the viewing window.

6. A method as described in claim 5, further comprising:

allocating the space within the viewing window by assigning first a dimension available in a primary axis of orientation indicated by said primary direction indicator to said regions in the order of the specified region priorities as indicated by said region priority indicators.

7. A method as described in claim 6, further comprising:

arranging said regions within said space in accordance with said relative location indicators of said regions.

8. A method as described in claim 7, further comprising:

allocating space in a direction orthogonal to said primary axis of orientation; and

arranging said regions within said window space in accordance with said relative location indicators.

* * * * *

US005371847A

United States Patent [19]

Hargrove

[11] **Patent Number: 5,371,847**

[45] **Date of Patent: Dec. 6, 1994**

[54] **METHOD AND SYSTEM FOR SPECIFYING THE ARRANGEMENT OF WINDOWS ON A DISPLAY**

[75] Inventor: **Richard R. Hargrove**, Redmond, Wash.

[73] Assignee: **Microsoft Corporation**, Redmond, Wash.

[21] Appl. No.: **949,401**

[22] Filed: **Sep. 22, 1992**

[51] **Int. Cl.**[5] **G06F 3/14**

[52] **U.S. Cl.** **395/157**; 395/161

[58] **Field of Search** 395/157, 158, 161, 155

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,961,070	10/1990	Maher et al.	395/158 X
5,001,697	3/1991	Torres	395/139 X
5,060,170	10/1991	Bourgeois et al.	395/157 X

OTHER PUBLICATIONS

Constraint-Based Tiled windows (E. S. Cohent, E. T. Smith, L. A. Iverson), 1984, pp. 2-11.

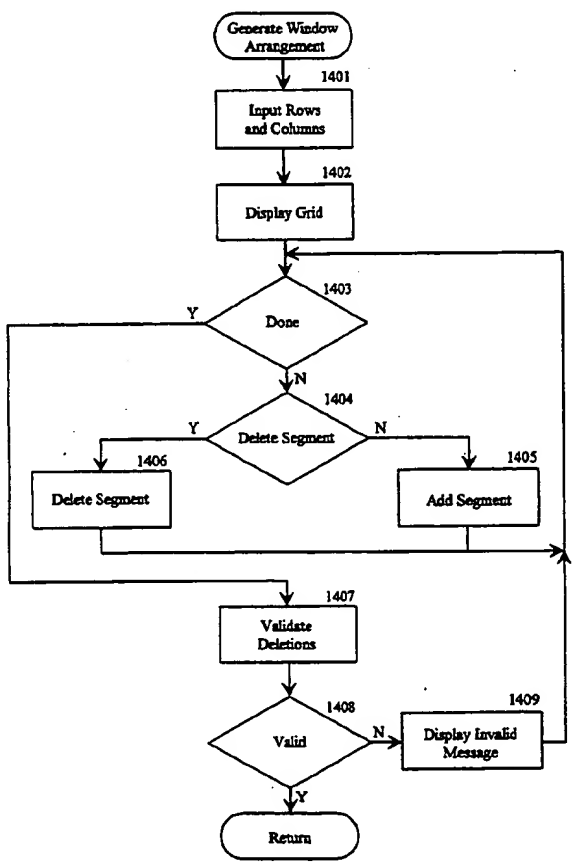
D. Raker & H. Rice, Inside Autocad, 1989 pp. Intro-1,7; 3-8, 3-9, 5-10, 5-11, 8-8.

Primary Examiner—Heather R. Herndon
Assistant Examiner—Cliff Nguyen Vo
Attorney, Agent, or Firm—Seed and Berry

[57] **ABSTRACT**

A method and system for specifying the arrangement of windows on a display device [is provided]. [In a preferred embodiment of the present invention, a] A selection grid is displayed on the display device. The selection grid has a bounding rectangle which represents the bounds of the display device and has a plurality of lines extending vertically and horizontally across the bounding rectangle. A user selects which line segments should be removed from the selection grid. A line segment is defined by the intersection points of the vertical and horizontal lines. As the user selects a line segment, the line segment is removed from the selection grid. The line segments that are not selected define the arrangement for the windows. The computer system then arranges the windows on the display in accordance with the specified window arrangement.

36 Claims, 12 Drawing Sheets



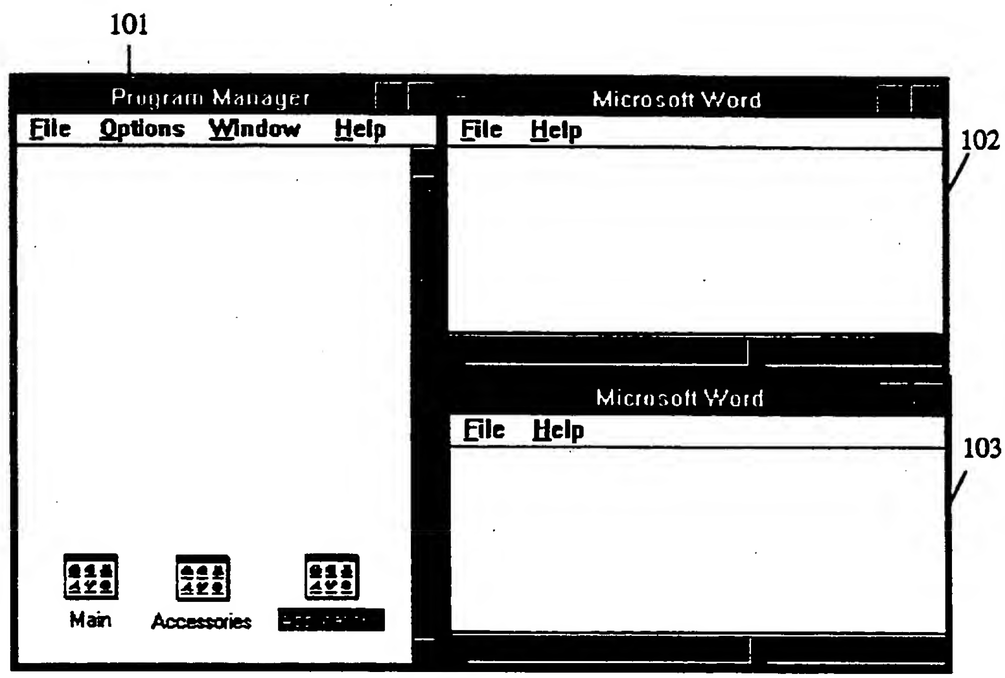


Figure 1
Prior Art

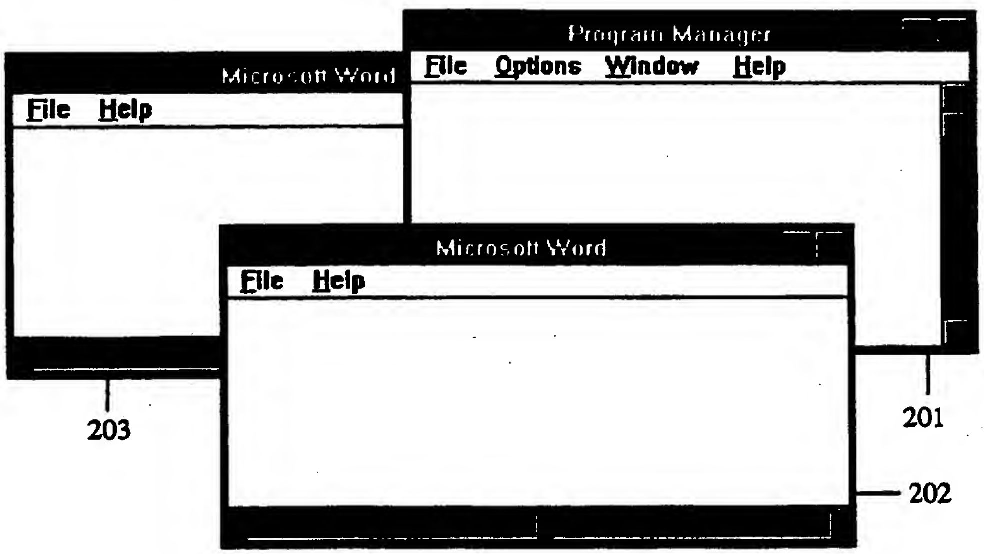


Figure 2
Prior Art

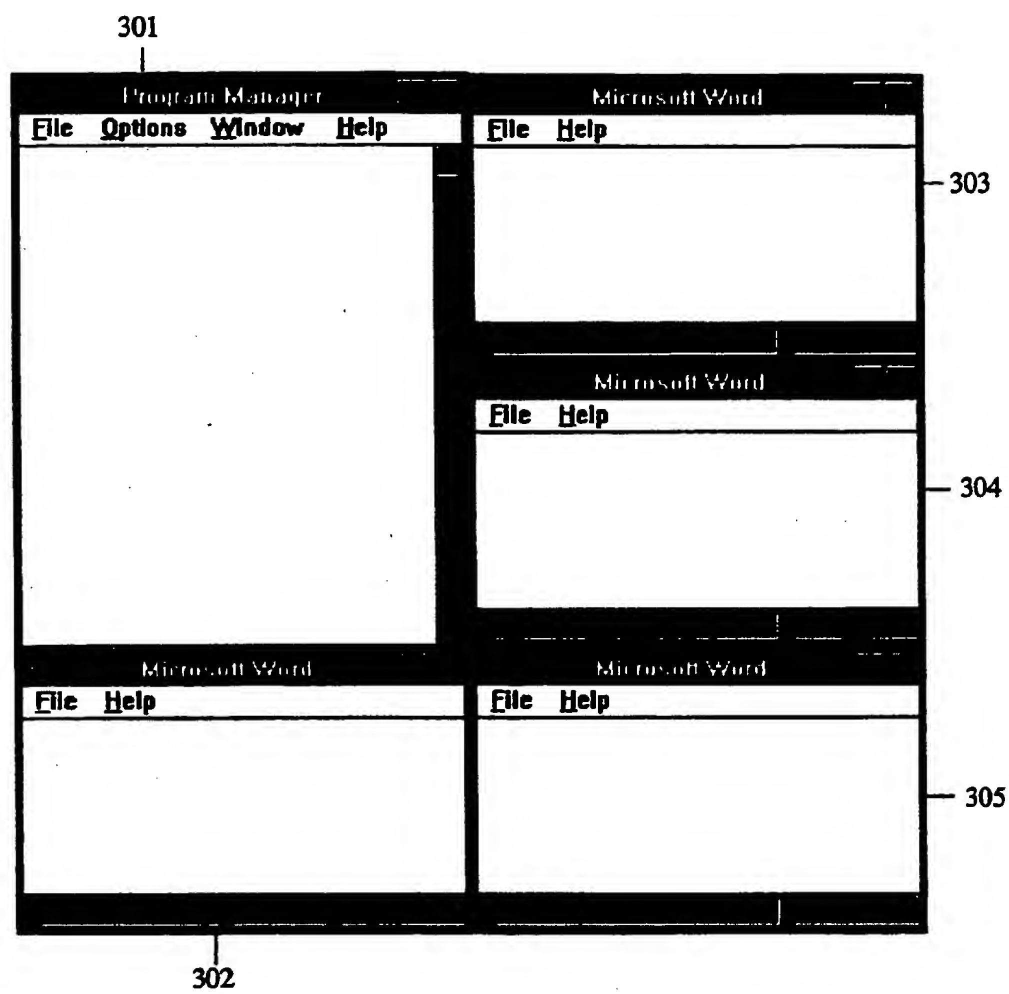


Figure 3
Prior Art

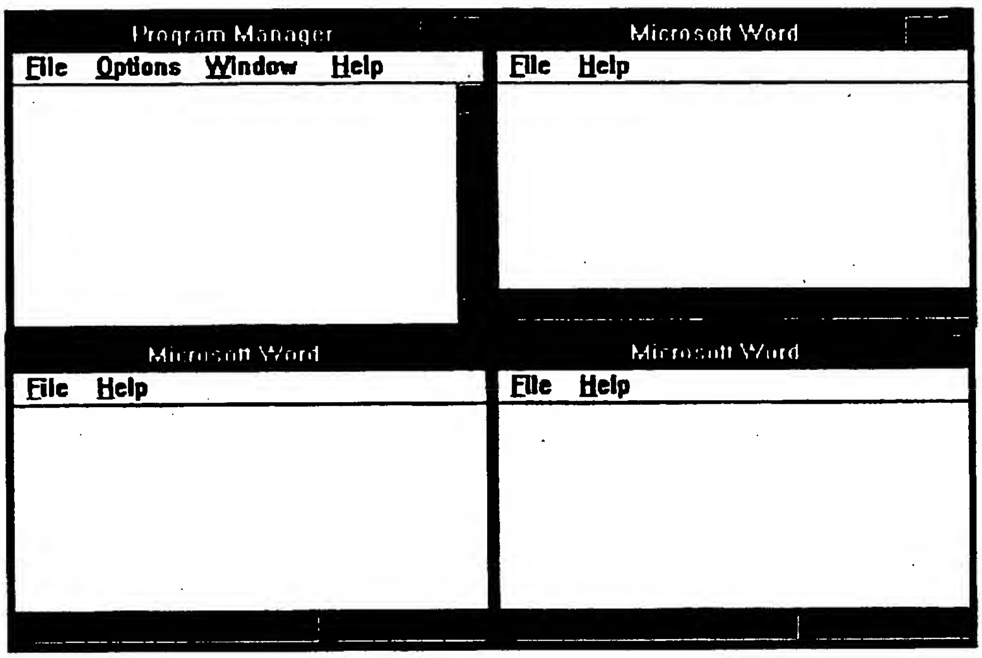


Figure 4

Prior Art

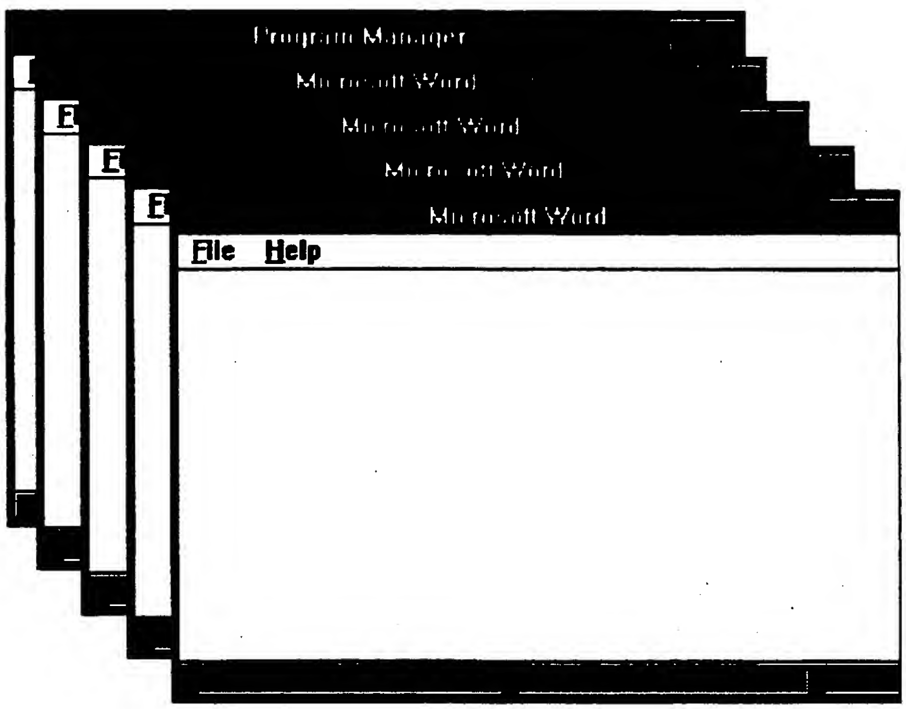


Figure 5
Prior Art

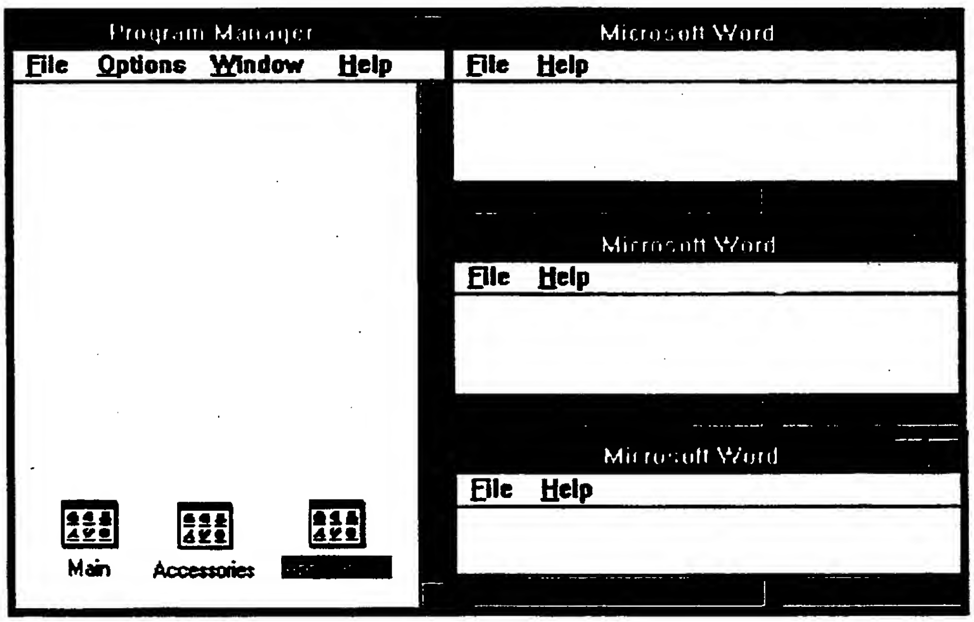


Figure 6
Prior Art

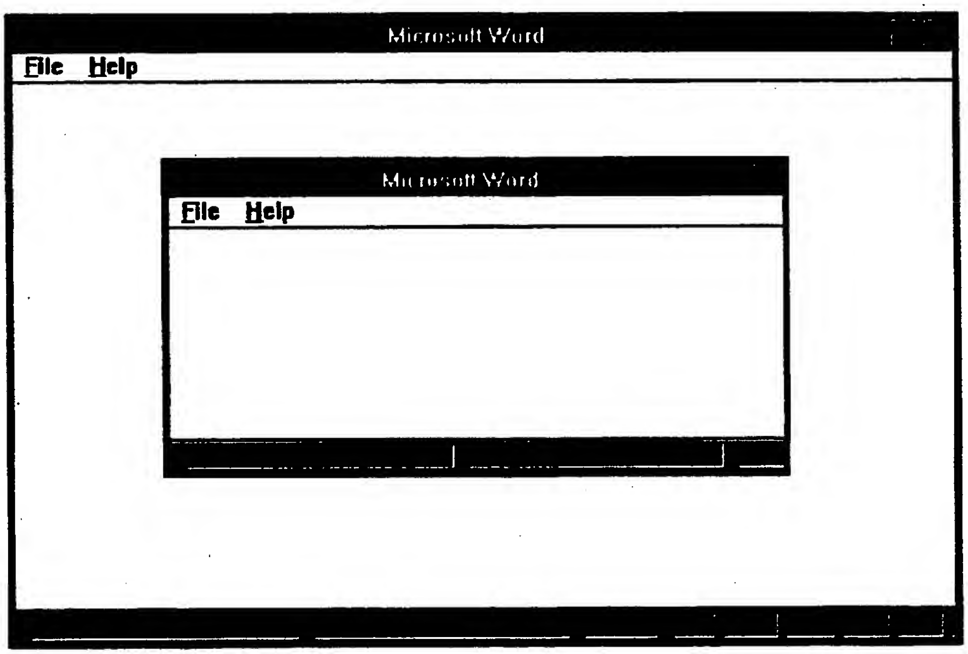


Figure 7
Prior Art

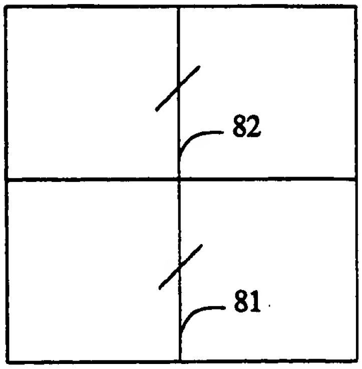


Figure 8A

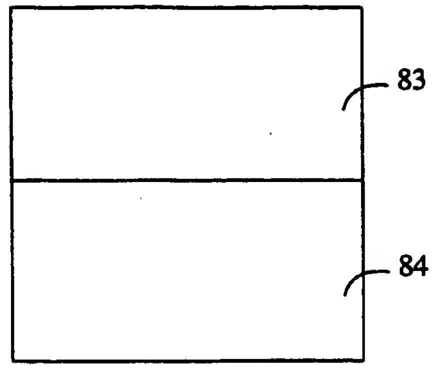


Figure 8B

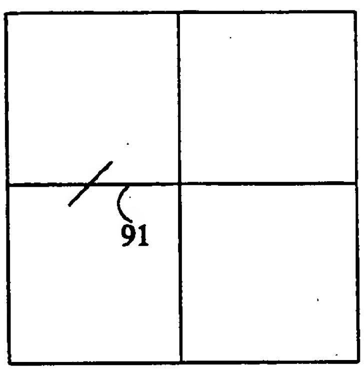


Figure 9A

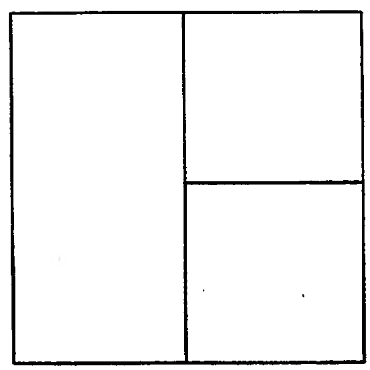


Figure 9B

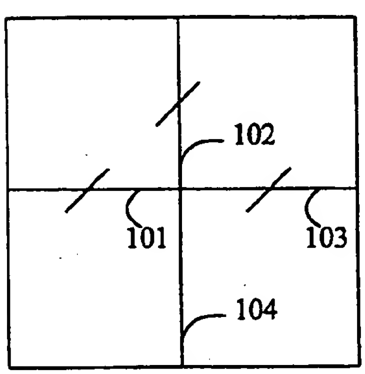


Figure 10A

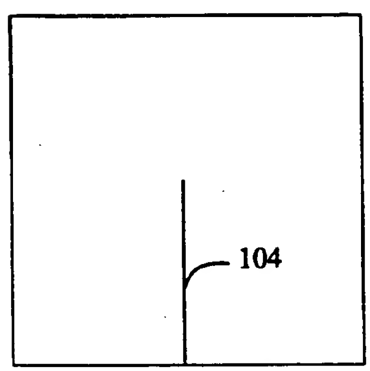


Figure 10B

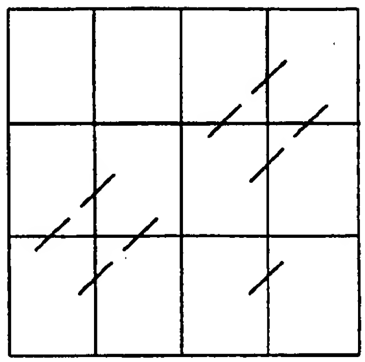


Figure 11A

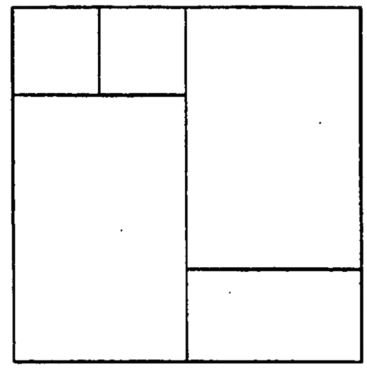


Figure 11B

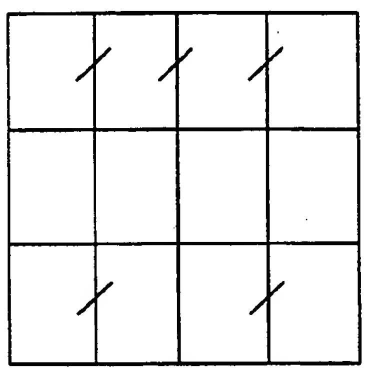


Figure 12A

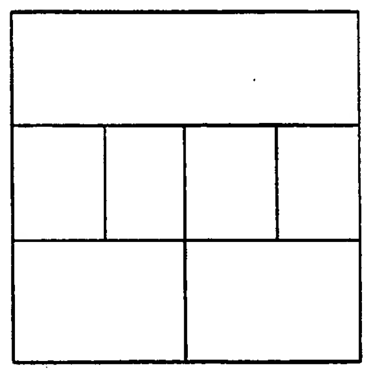


Figure 12B

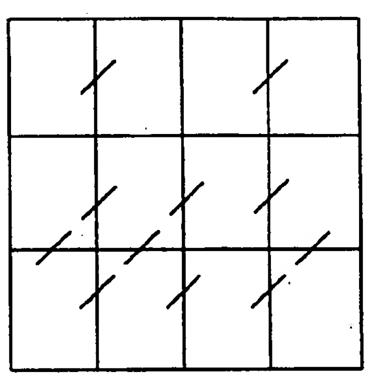


Figure 13A

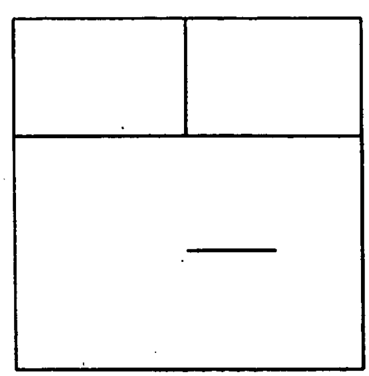
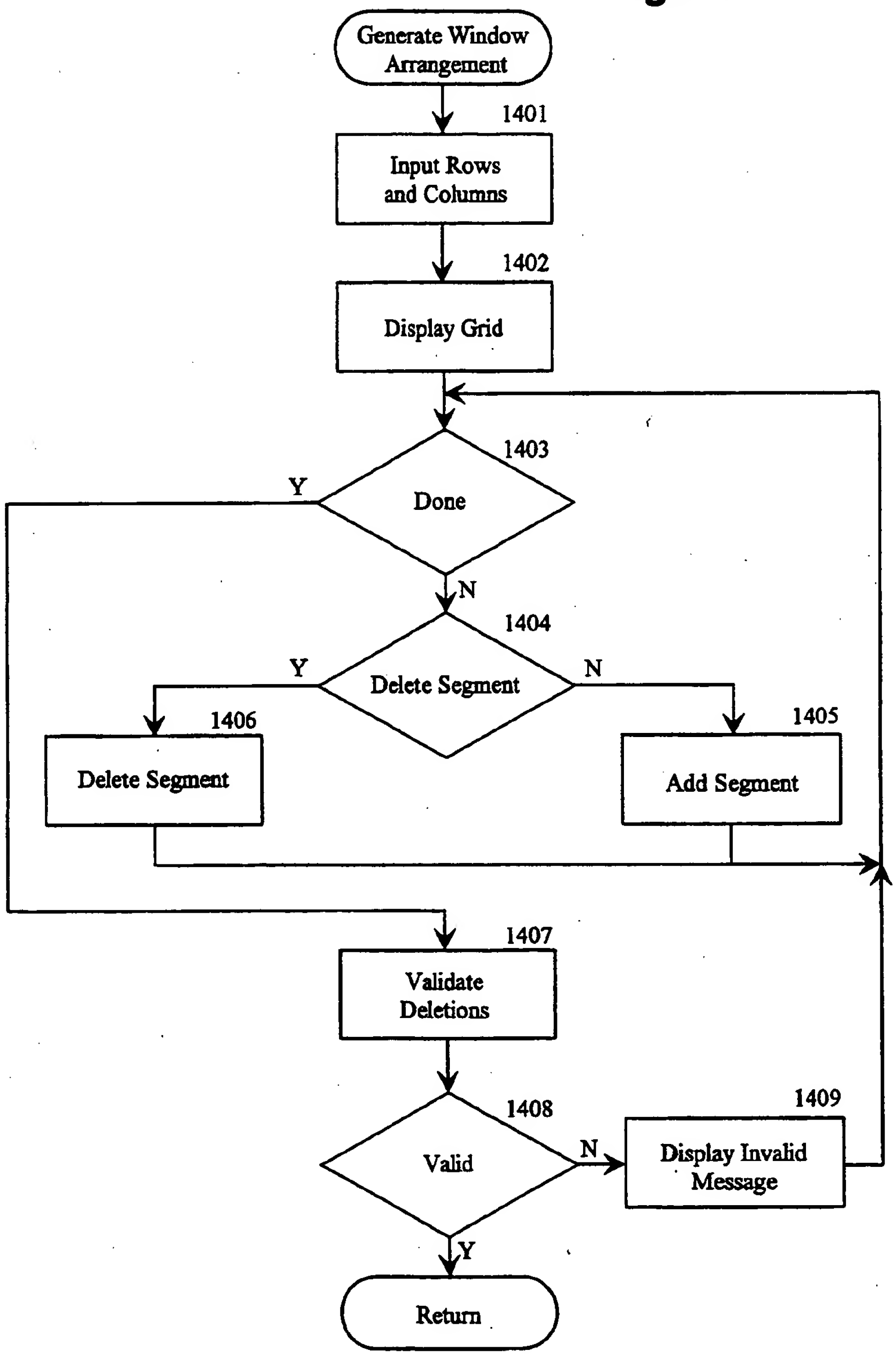


Figure 13B

Figure 14


Generate Window Arrangement
1401
Input Rows and Columns
1402
Display Grid
1403
Done
Y
N
1404
Delete Segment
Y
N
1406
Delete Segment
1405
Add Segment
1407
Validate Deletions
1408
Valid
N
1409
Display Invalid Message
Y
Return

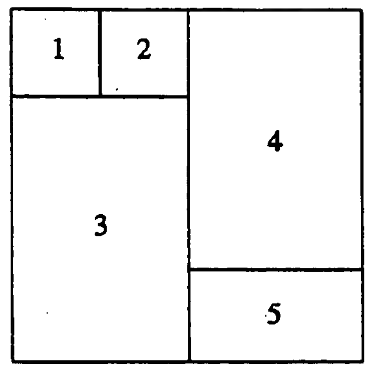


Figure 15

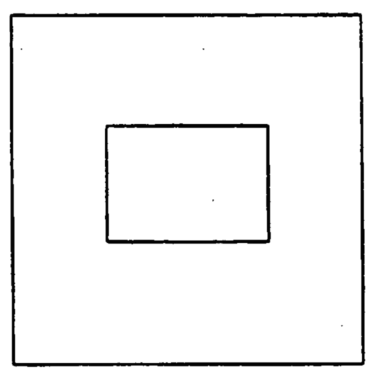


Figure 16

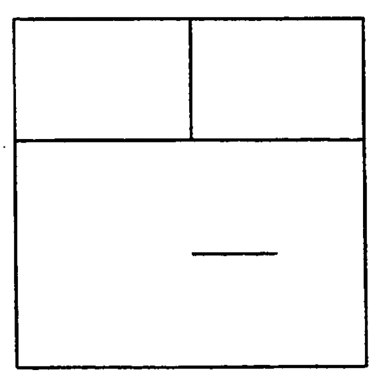


Figure 17

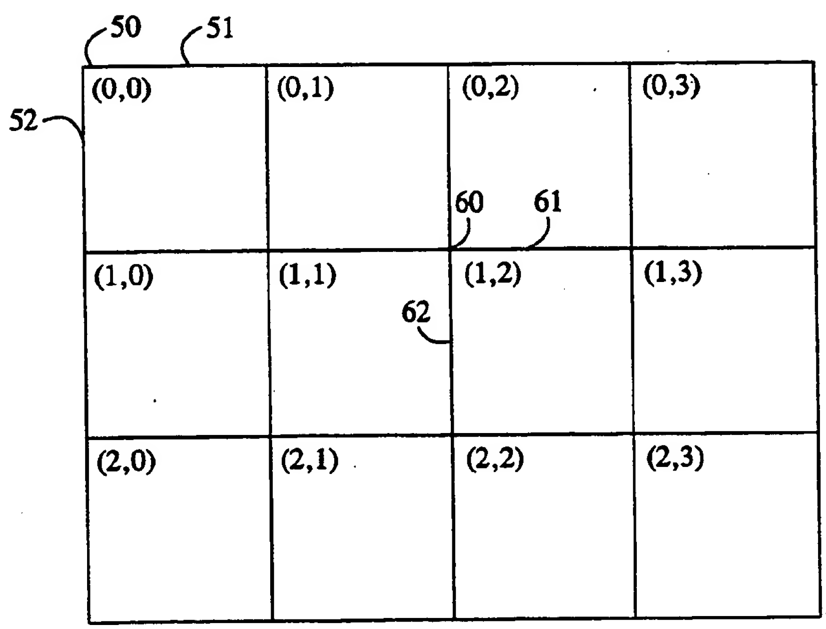


Figure 18

METHOD AND SYSTEM FOR SPECIFYING THE ARRANGEMENT OF WINDOWS ON A DISPLAY

DESCRIPTION

1. Technical Field

This invention relates generally to a computer method and system for displaying information, and more specifically, a method and system for specifying the arrangement of windows on a display.

2. Background of the Invention

Information generated by a computer program is often displayed on a display device. When multiple programs display information simultaneously on the same display device, a rectangular area of the display is typically allocated to each program. These rectangular areas are referred to as windows. The use of a window helps a user to identify which program generated the displayed information. A window typically includes a border to delimit the bounds of the window and a title bar to identify the program that is sending information to the window.

FIG. **1** is a diagram of a display showing three windows. The three windows, **101**, **102**, and **103**, are in a tiled arrangement; that is, the windows are side-by-side and do not overlap one another. Each of the three computer programs can send data to the windows concurrently so that the data is displayed simultaneously. FIG. **2** is an example of overlapping windows. Window **201** overlaps window **203**, and window **202** overlaps both window **201** and window **203**. Most computer systems require that a window be rectangular. Computer systems allow for considerable flexibility in the arrangement of multiple windows on the display. FIG. **3** shows five windows in a tiled arrangement. A user can spend considerable time in arranging the windows. For example, to generate the window arrangement of FIG. **3**, a user would need to individually size and move each window. To facilitate the arranging of windows, windowing systems, such as Windows by Microsoft Corporation, provide means for automatically arranging windows. These windowing systems allow for windows to be automatically arranged in a tiled arrangement. In response to requests to automatically generate a tiled arrangement with five windows, the windowing systems typically generate a window arrangement as shown in FIG. 3. FIG. **4** shows an automatically generated tiled arrangement for four windows.

Windowing systems also allow for windows to be automatically arranged in a cascaded arrangement. FIG. **5** shows five windows arranged in cascaded arrangement. Although cascading can be done quite efficiently when done automatically by the windowing system, it would be very time consuming for a user to size and move each window to arrange them as shown in FIG. **5**.

Although the automatic cascading and tiling of windows relieves the user from the burden of sizing and moving windows, the windows are only automatically arranged in certain predefined arrangements. For example, when four windows are to be arranged, the automatic tiling will typically arrange the windows as shown in FIG. **4**. If a user wants four windows arranged as shown in FIG. **6**, then the user must manually size and move each window. Also, the only overlapping of windows that is automatically performed by a typical windowing system is the cascading of the windows. FIGS. **2** and **7** show overlapping windows that are not cascaded. A user would need to size and move each window manually to generate these arrangements.

Studies have been conducted to determine whether a window or a non-window environment provides a more efficient user interface. These studies have shown that tasks often take longer to complete in a window environment than in a non-window environment. The users in a window environment spend considerable time sizing and moving the windows and scrolling the windows to bring necessary information into view. These studies have also shown that, after eliminating the time to arrange the windows, the tasks were accomplished quicker in the window environment. Consequently, many window environments provide support for the automatic arrangement of windows.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a method and system for specifying the arrangement of windows on a display device.

It is another object of the present invention to provide a method and system for arranging windows on a display device in accordance with a user-specified arrangement.

It is another object of the present invention to provide a method and system for specifying a window arrangement that allows windows to be tiled.

It is another object of the present invention to provide a method and system that combines flexibility in arranging windows and efficiencies associated with automatic arrangement.

These and other objects, which will become apparent as the invention is more fully described below, are obtained by an improved method and system for specifying the arrangement of windows on a display device. In a preferred embodiment of the present invention, a selection grid is displayed on a display device. The selection grid has a bounding rectangle, which represents the bounds of the display device, and has a plurality of lines extending vertically and horizontally across the bounding rectangle. A user selects which line segments should be removed from the selection grid. A line segment is defined by the intersection points of the vertical and horizontal lines. As the user selects a line segment, the line segment is removed from the selection grid. When the user has completed selecting line segments, the line segments that are not removed define the arrangement for the positioning of windows within the main window of an application program. The computer system then arranges the windows on the display in accordance with the specified window arrangement.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a display of three windows in a tiled arrangement.

FIG. **2** is a display of three windows in an overlapped arrangement.

FIG. 3 is a display of five windows in a tiled arrangement.

FIG. 4 is a display of four windows in a tiled arrangement.

FIG. 5 is a display of five windows in a cascaded arrangement.

FIG. 6 is a display of four windows in a manually-generated tiled arrangement.

FIG. 7 is a display of two windows in a manually-generated overlapped arrangement.

FIG. 8A through 13B show sample grid selections and resulting window arrangements.

FIG. 14 is a flow diagram of a routine to input a user window arrangement selection.

FIG. 15 is a display of a valid window arrangement.

FIGS. 16 and 17 are displays of invalid window arrangements.

FIG. 18 is a diagram illustrating data structures used by a window layout validity algorithm.

DETAILED DESCRIPTION OF THE INVENTION

The present invention provides a method and system for a user to specify the arrangement of windows on a display of a computer system. In a preferred embodiment of the present invention, a user specifies a window arrangement from a selection grid on the display. The user preferably inputs the number of rows and columns that the grid is to contain. The user then selects which line segments should be removed from the grid. The remaining line segments define the window arrangement.

FIGS. 8A through 13B show sample grid selections and resulting window arrangements. FIG. 8A shows a grid comprising two rows and two columns. The grid contains four quadrants. The user selects which line segments to remove from the grid to give the desired window arrangement. For example, in FIG. 8A, the slash across line segments 81 and 82 indicates that the user selects to remove line segments 81 and 82 from the grid. The resulting window arrangement is shown in FIG. 8B. In a preferred embodiment of the present invention, a user uses a pointing device, such as a mouse, to specify which line segments to remove from the grid. For example, in FIG. 8A when the user clicks on line segment 81, that line segment would be removed from the grid and when the user clicks on line segment 82, it would also be removed from the grid leaving the arrangement of FIG. 8B. It is preferable that if the user then reclicks at approximately where line segment 81 or 82 was, then the corresponding line segment would be redisplayed. This would allow the user to revise the selection grid. When the user has removed the appropriate line segments, the methods of the present invention would then arrange the windows on the display to correspond to the selected arrangement. For example, in FIG. 8B the display would comprise an upper window 83 and a lower window 84. If there were more than two open windows in the system, then the methods of the present invention would determine which two windows to display in windows 83 and 84. Conversely, if there was only one open window in the system, then either window 83 and 84 would be displayed, but one would be left empty.

FIG. 9 shows the selection of a window arrangement corresponding to FIG. 1. FIGS. 9A and 9B show the selection of line segment 91, which results in the window arrangement shown in FIG. 9B.

Not all selections of line segments result in valid window arrangements. The selection grid shown in FIG. 10A results in the invalid window arrangement request of FIG. 10B. In FIG. 10A the user selected line segments 101, 102, and 103. The resulting window arrangement is shown in FIG. 10B with line segment 104. Since the window is defined to be a rectangular area and since line segment 104 is not part of a rectangular area, the resulting window arrangement is invalid. It is preferable that the system of the present invention displays a warning message to the user and allows the user to either add another line segment or remove line segment 104.

The method and system of the present invention preferably allows the user to specify the number of rows and columns to display on the grid. For example, in FIGS. 11A, 12A, and 13A the user specifies that the grid is to contain three rows and four columns. FIGS. 11A, 12A, and 13A show various window arrangements that the user may specify using a 3×4 grid.

FIG. 14 is a flow diagram of a routine to allow a user to specify a window arrangement. In step 1401, the routine inputs the number of rows and columns from the user. Alternatively, the routine uses a default number of rows and columns and allow the user to override these defaults. In step 1402, the routine displays a selection grid with the specified number of rows and columns. In steps 1403 through 1406, the routine loops, allowing the user to select and deselect the displayed line segments. In step 1403, if the user specifies that the selection is complete, the routine then continues at step 1407, else the routine continues at step 1404. In step 1404, the routine determines whether the selected line segment is to be removed from the selection grid or added back to the selection grid. In a preferred embodiment, when the user clicks near a line segment, that segment is removed from the selection grid. Conversely, if the user clicks near a removed line segment, then that line segment is added back to the selection grid. In an alternate embodiment, the routine allows a user to remove multiple line segments by holding down the mouse and dragging the cursor across various line segments. As the cursor crosses a line segment, the line segment is removed. In step 1404, if a line segment is to be removed, then the routine continues at step 1405, else the routine continues at step 1406. In step 1405, the routine removes the selected segment from the selection grid and loops back to step 1403. In step 1406, the routine adds the removed line segment back onto the selection grid and loops back to step 1403. In step 1407, the routine ensures that the user has specified a valid window arrangement. In step 1408, if the window arrangement is valid, then the window arrangement selection is complete and the routine returns, else the routine continues at step 1409. In step 1409, the routine displays an invalid window arrangement message and loops to 1403 to allow the user to correct the window arrangement.

A window arrangement is valid if each rectangle formed by the remaining line segments either (1) has at least one line that traverses it horizontally or vertically, or (2) has no line segments within it. FIG. 15 is a display of a valid arrangement. The window arrangement of FIG. 15 defines 9 rectangles. Rectangles 1, 2, 3, 4, and 5 contain no line segments. Rectangles 4 and 5 form a larger rectangle that has a line that traverses it. Rectangles 1 and 2 form a larger rectangle that has a line that traverses it. Rectangles 1, 2, and 3 form a rectangle that has a line that traverses it. Rectangles 1, 2, 3, 4, and 5 form a rectangle the size of the screen that has a line that traverses it. Since each of these 9 rectangles has a line that traverses it or no line segments within it, then the window arrangement is valid.

FIGS. 16 and 17 are displays of invalid window arrangements. In FIG. 16, since the rectangle corresponding to the perimeter of the screen has no traversing line, the window arrangement is an invalid arrangement. In FIG. 17, since the lower rectangle contains a line seg-

ment but has no line that traverses it, the window arrangement is invalid.

TABLE 1

```
for c = 0, cColumns-1
   for r = 0, cRows-1
         if (Point[r,c].south is not removed and
            Point[r,c].east is not removed) then
              If (~ValidRect(r, c)) then
                    Invalid Arrangement
              endif
         endif
   endfor
endfor
Valid Arrangement

ValidRect(r, c)
   ValidRect = FALSE
   Find the bottom-left corner
   Find the top-right corner
   if (segment missing on bottom or on right) then Return
   if (no segments within rectangle) then ValidRect=TRUE
   Return
```

Table 1 list pseudocode for an algorithm that determines whether a window arrangement is valid. FIG. **18** is a diagram illustrating data structures used by the algorithm. The upper-left corner of each rectangle in the selection grid is a point. Each point is identified by row and column number. For example, point **50** is identified by row number **0** and column number **0** (0,0), and point **60** is identified by row number **1** and column number **2** (1,2). The information specifying the window arrangement is stored in array Point. Array Point is indexed by row and column number and contains two fields: south and east. These fields indicate whether the line segment below (south) and the line segment to the right (east) of the point have been removed. Line segment **51** is east of point **50**, and line segment **52** is south of point **50**. Line segment **61** is east of point **60**, and line segment **62** is south of point **60**. If line segment **61** were removed, then Point[1,2].east would indicate removed and Point[1,2].south would indicated not removed.

The algorithm determines validity of a window arrangement as follows. The algorithm loops selecting each point. If the selected point has both the south and east line segments present, then the point may be an upper-left corner of a valid rectangle and the algorithm determines rectangle validity, otherwise the algorithm loops to select the next point. The algorithm determines rectangle validity by first searching for the lower-left corner. The algorithm checks each point below the selected point until if finds a point with a east line segment, which indicates a corner. If the algorithm encounters a point with a south line segment removed before encountering a point with an east line segment, then the rectangle is invalid. The algorithm then searches for the upper-right corner. The algorithm checks each point to the right of selected point until if finds a point with a south line segment, which indicates a corner. If the algorithm encounters a point with a east line segment removed before encountering a point with a south line segment, then the rectangle is invalid. Finally, if there are line segments within the rectangle, then the rectangle is invalid. Otherwise, the algorithm loops to select the next point. The window arrangement is valid if all points are processed without encountering an invalid rectangle.

Although the methods and systems of the present invention have been disclosed and described in terms of a preferred embodiment, it is not intended that the present invention be limited to such embodiments. Modifications within the spirit of the invention will be apparent to those skilled in the art. The scope of the present invention is defined by the claims that follow.

I claim:

1. A method in a computer system for specifying a display arrangement for a plurality of windows, the computer system having a display device, the method comprising the steps of:

displaying on the display device a selection grid, the selection grid having a bounding rectangle with a height and a width, and having a vertical line extending the height of the bounding rectangle and a horizontal line extending the width of the bounding rectangle, the vertical and horizontal lines intersecting to form a plurality of line segments; and

selecting a plurality of the line segments wherein the bounding rectangle and the line segments that are not selected define the display arrangement for the plurality of windows.

2. The method of claim **1** including the step of after selecting a line segment, removing the selected line segment from the display device.

3. The method of claim **2** including the step of displaying a previously removed line segment in response to an indication that the previously removed line segment should be deselected.

4. The method of claim **1** wherein a line segment is selected using a pointing device.

5. The method of claim **4** wherein the painting device has a button and a cursor is displayed on the device and wherein a plurality of line segments is selected by depressing the button, moving the cursor across line segments to be selected, and releasing the button wherein the line segments that the cursor moved across while the button was depressed are selected.

6. The method of claim **4** wherein the pointing device is a mouse.

7. The method of claim **1** including the step of receiving from a user an indication of a number of vertical and a number of horizontal lines to include in the selection grid.

8. The method of claim **7** wherein the indication is a number of rows and columns to be displayed in the selection grid, from which the number of vertical and the number of horizontal lines is derived.

9. A method in a computer system for arranging a plurality of windows on a display device, each window having a display geometry indicating location and size of the window on the display device, the method comprising the steps of:

displaying on the display device a selection grid, the selection grid having a bounding rectangle with a height and a width, and having a vertical line extending the height of the bounding rectangle and a horizontal line extending the width of the bounding rectangle, the vertical and horizontal lines intersecting to form a plurality of line segments;

selecting a plurality of the line segments wherein the bounding rectangle and the line segments that are not selected define a display arrangement for the plurality of windows; and

defining the geometry of the plurality of windows to correspond to the geometry of the defined display arrangement.

10. The method of claim **9** including the step of after selecting a line segment, removing the selected line segment from the display device.

11. The method of claim 10 including the step of displaying a previously removed line segment in response to an indication that the previously removed line segment should be deselected.

12. The method of claim 9 including the step of receiving from a user an indication of a number of vertical and a number of horizontal lines to include in the selection grid.

13. The method of claim 12 wherein the indication is a number of rows and columns to be displayed in the selection grid, from which the number of vertical and the number of horizontal lines is derived.

14. A method in a computer system for specifying a display arrangement for a plurality of windows, the computer system having a display device, the method comprising the steps of:

displaying on the display device a selection grid, the selection grid having a bounding rectangle and having a plurality of lines extending across the bounding rectangle; and

selecting a line wherein the bounding rectangle and the lines that are not selected define the display arrangement for the plurality of windows.

15. The method of claim 14 including the step of after selecting a line, removing the selected line from the display device.

16. The method of claim 15 including the step of redisplaying a previously removed line in response to an indication that the previously removed line should be deselected.

17. A computer system for specifying a display arrangement for a plurality of windows comprising:

a computer;

a display device operatively connected to the computer for displaying a plurality of windows;

means for displaying on the display device a selection grid, the selection grid having a bounding rectangle with a height and a width, and having a vertical line extending the height of the bounding rectangle and a horizontal line extending the width of the bounding rectangle, the vertical and horizontal lines intersecting to form a plurality of line segments; and

means for selecting a plurality of the line segments wherein the bounding rectangle and the line segments that are not selected define the display arrangement for the plurality of windows.

18. The computer system of claim 17 including means for removing selected line segments from the display device.

19. The computer system of claim 18 including means for deselecting a previously selected line segment and means for displaying a deselected line segment.

20. The computer system of claim 17 including a pointing device for selecting a line segment.

21. The computer system of claim 17 including a mouse for selecting a line segment.

22. The computer system of claim 17 including means for indicating a number of vertical lines extending the height of the selection grid and a number of horizontal lines extending the width of the selection grid.

23. A method in a computer system for arranging windows on a display device, each window having a display geometry indicating location and size of the window on the display device, the method comprising the steps of:

receiving from a user an indication of a number of rows and columns of windows to include in a display grid;

developing a window arrangement wherein the windows are to be arranged in accordance with the received number of rows and columns; and

adjusting the geometry of a plurality of windows to correspond to the developed window arrangement.

24. A method in a computer system for specifying a display arrangement for a plurality of windows, the computer system having a display device, the method comprising the steps of:

displaying on the display device a selection grid, the selection grid having a bounding rectangle, and having a plurality of lines extending from a side of the bounding rectangle to another side of the bounding rectangle; and

selecting one of the plurality of lines to exclude from the selection grid, wherein the bounding rectangle and lines that are not selected define the display arrangement for the plurality of windows.

25. The method of claim 24 including the step of, after selecting one of the plurality of lines, removing the selected line from the display device.

26. The method of claim 25 including the step of displaying a previously removed line in response to an indication that the previously removed line should be deselected.

27. The method of claim 24 wherein the line is selected using a pointing device.

28. The method of claim 27 wherein the pointing device has a cursor to indicate motion, and wherein lines are selected by dragging the pointing device and moving the cursor across lines to be selected, wherein each line that the cursor moves across while dragging the pointing device is selected.

29. The method of claim 27 wherein the pointing device is a mouse.

30. A method in a computer systems for specifying a display arrangement for a plurality of windows, the computer system having a display device, each window having a window geometry with lines defining the size of the window and with a position defining the location of the window on the display device, the method comprising the steps of:

altering the window geometry of a window displayed on the display device;

displaying on the display device a selection grid, the selection grid having a bounding rectangle, and having a plurality of lines distinct from the lines of the window geometries of the plurality of windows, wherein the plurality of lines of the selection grid symbolically represents the lines of the plurality of windows as altered, and wherein the intersecting lines of the selection grid form a plurality of line segments; and

selecting a plurality of line segments to exclude from the selection grid, wherein the bounding rectangle and the line segments that are not selected define the display arrangement for the plurality of windows.

31. The method of claim 30, further including the step of adjusting the window geometries of the plurality of windows according to the defined display arrangement.

32. A method in a computer systems for specifying a display arrangement for a plurality of windows, the computer system having a display device, each window

having a window geometry with lines defining the size of the window and with a position defining the location of the window on the display device, the method comprising the steps of:

displaying on the display device a selection grid, the selection grid having a bounding rectangle, and having a plurality of lines distinct from the lines of the window geometries of the plurality of windows, wherein the plurality of lines of the selection grid symbolically represents the lines of the plurality of windows, and wherein the intersecting lines of the selection grid form a plurality of line segments; and

selecting a plurality of line segments to exclude from the selection grid wherein the bounding rectangle and the line segments that are not selected define the display arrangement for the plurality of windows.

33. A computer system for specifying a display arrangement for a plurality of windows, each window having a window geometry with lines defining the size of the window and with a position defining the location of the window, the computer system comprising:

a display device for displaying a selection grid, the selection grid having a bounding rectangle, and having a plurality of lines distinct from the lines of the window geometries of the plurality of windows, wherein the plurality of lines of the selection grid symbolically represents the lines of the plurality of windows, and wherein the intersecting lines of the selection grid form a plurality of line segments;

an input device for receiving a selection of a plurality of line segments to exclude from the selection grid; and

a window arrangement definition mechanism for defining the display arrangement for the plurality of windows based upon the bounding rectangle and the line segments that are not selected.

34. The computer system of claim 33, further comprising a window arranger for arranging the window geometries of the plurality of windows in accordance with the defined display arrangement.

35. A computer system for specifying a display arrangement for a plurality of windows comprising:

a selection grid having a bounding rectangle and having a plurality of lines extending across the bounding rectangle;

a display device for displaying the selection grid;

an input device for selecting a line;

a display arrangement definition mechanism for defining the display arrangement for the plurality of windows using the bounding rectangle of the selection grid and the lines that are not selected.

36. The computer system of claim 35 wherein the selection grid, responsive to the selection of a line using the input device, causes the selected line to be removed from the display device.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,371,847

DATED : December 6, 1994

INVENTOR(S) : Richard R. Hargrove

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In column 6, claim 5, line 29, please delete "painting" and substitute therefor --pointing--.

In column 6, claim 5, line 30, after "on the" and before --device--, please insert "display".

Signed and Sealed this

Eighteenth Day of April, 1995

Attest:

Bruce Lehman

BRUCE LEHMAN

Attesting Officer

Commissioner of Patents and Trademarks